Glass OTP: Secure and Convenient User Authentication on Google Glass

Pan Chan, Tzipora Halevi, Nasir Memon

NYU School of Engineering, New York, USA
{pc1260,thalevi,memon}@nyu.edu

Abstract. Wearable computing devices have become increasingly popular and while these devices promise to improve our lives, they come with new challenges. This paper focuses on user authentication mechanisms for the Google Glass device (Glass). Glass only has three sources of input: a camera, a microphone, and a touchpad. This limited set of interfaces makes the use of standard passwords infeasible or cumbersome. We therefore propose a One-Time-Password (OTP) authentication scheme, Glass OTP that uses the Glass camera to scan a QR code displayed on the user's smartphone. We implement a proof of concept Glass lock screen which unlocks only upon scanning an OTP generated by the companion Android smartphone application. We also discuss the reliability, security, and convenience of the proposed solution as compared to the current solutions in use.

Keywords: authentication, user authentication, wearable computing, google glass, one time passwords, password alternative

1 Introduction

As technology advances and miniaturizes, companies have begun to innovate by creating new wearable computing devices ranging from simple bracelets for tracking daily activity [1] to reality augmenting head-mounted displays. The focus of this paper is on the latter type of wearable device, specifically the Glass by Google [2].

Glass presents new challenges in terms of security. Being a device worn on a user's head, Glass is potentially an easier target for thieves. In addition, the limited input capabilities available make the creation of an authentication mechanism that balances security with usability much more difficult. This paper proposes Glass OTP, a simple authentication mechanism for the Glass and discusses its security challenges and considerations of possible solutions.

As of the time of this paper's writing, two authentication applications exist for the Glass. The first and oldest is "Bulletproof" released in April 2013 [7]. The second and most recent one is the lock screen developed by Google and included as of version XE12 [8]. Both of these authentication mechanisms are similar in that they interpret a user's gestures on the touchpad of the Glass to unlock the device.

Bulletproof is a relatively simple lock screen, developed and released as the first lock screen for Glass. It allows a user to configure a combination of single tap, long tap, fling left and fling right to unlock the device. Fling left is a swipe from the back of a user's head to the front, fling right is the opposite. The combination chosen by a user is written in the source code before the app is built; as a result there is no limit on how long a combination can be [7].

The official Glass lock screen was released and came packaged with the XE12 update in December 2013. This lock screen works similarly to Bulletproof, but has more gesture options and is a more polished product. Available gestures include swiping forwards or backwards with one or two fingers, hook swipes – swiping forward and back in the same motion, and tapping with one or two fingers. The configuration for this lock screen is built into the settings UI of Glass. A user is required to use a combination of exactly four gestures. In the case of a forgotten combination, a user may log in to the MyGlass website and obtain a QR code to reset the Glass lock screen [8].

The touchpad based method of authentication described above aims to be intuitive and easy, but has some limitations in terms of security. In Google's implementation, there are 4096 possible lock combinations and also a lock out after enough failed attempts. Bulletproof does not have a lock out threshold, but it does allow for a technically unlimited combination length. As a result, both of these implementations appear to mitigate random brute force attacks.

However, there is one major issue that can make these implementations insecure. The unlock pattern combination must be entered by a user on the touchpad which is located by the user's right temple. This means that the gesture a user enters is very visible to all surrounding people. Unlike traditional "shoulder surfing" attacks, an attacker observing a Glass user from a far is much less obvious and suspicious. This also means that it is very easy for a person to record a user unlocking their Glass from a distance without arousing suspicion. A user can also be unintentionally recorded by surveillance equipment in the vicinity.

Usability of touchpad gesture based unlock patterns on the glass is also of limited value. In a recent study that proposed a novel touch-based pin (TBP) authentication [22], 27% of the participants achieved successful login without any errors in 10 trials with a success rate of 87% overall for the method. The TBP method did surpass the Google Glass Pattern Lock mechanism, for which only 7% of the participants managed to log-in 10 times without any errors and provided 68% overall success rate. In that study, users indicated they preferred a voice-based pin authentication over TBP, and that both options were preferred over the Glass Pattern Lock mechanism.

The rest of this paper is organized as follows. In the next section, we explore different alternatives that one may employ to create a user authentication mechanism for Glass. In section 3 we present Glass OTP, a One-Time-Password (OTP) authentication scheme that can be used to secure the Glass. Two proof of concept applications that are developed and described: "Glass OTP Companion" application for Android smartphone devices and "Glass OTP" lock screen for Glass devices. In section 4 we analyze the security and usability of the proposed scheme based on a framework developed by Bonneau et. al. [9]. In section 5 we conclude with a discussion and a description of possible future work.

2 Options for user authentication on Glass

In simpler terms, Glass is a wearable device consisting of a small display, which can project a transparent picture in a person's line of view when worn on the head like a normal glass. It has a microphone, a camera, and a touchpad for input. Wi-Fi and Bluetooth are available for connectivity [2].

Glass is a device worn on the head during use and often worn for extended periods of time [2]. As a result, the Glass is potentially an easier target for theft. It is easy to identify someone who is wearing Glass and easy to steal Glass from the head of an inattentive user. With a higher risk of physical theft, the security of the data it contains becomes even more important and hence the need for user authentication.

If we consider the user input mechanisms available on the Glass, we have limited options for user authentication schemes. The standard keyboard based password or pin based authentication implemented in virtually all computing devices are infeasible to implement here as there is no physical keyboard for input, and it appears to be difficult to implement a usable virtual keyboard with the existing Glass hardware. This requires developers to get creative with the hardware provided.

Using the microphone, a user could speak a specific passphrase to unlock the Glass. This would leverage the existing development on voice recognition for Glass [3]. However, the obvious problem here would be an attacker eavesdropping for the passphrase. A novel solution to this has been recently proposed by Yadav et al. [22]. The proposed algorithm employs a pin substitution that is only visible to the user, therefore overcoming the threat from an eavesdropping attack.

Using the touchpad, a user could perform a pattern of gestures to unlock the Glass. There are two existing implementations based on this idea, one being the current official lock screen developed by Google [7, 8]. In section 4 will briefly analyze these existing solutions and their weaknesses.

Aside from traditional user-input, Bluetooth connectivity is available and may be used to transfer data between Glass and a user's smartphone. The use of Bluetooth as a proximity-based locking mechanism was considered prior to the solution presented in this paper. However, there were a few concerns that made Bluetooth an undesirable choice. Bluetooth is often interpreted as a major cause of battery drain and while Bluetooth 4.0's Low Energy mode greatly reduces power consumption [12], it must also be supported by the user's smartphone. For Android, Bluetooth 4.0 LE was not supported until the release of Android 4.3 (API 18) in June 2013 [13]. Additionally, Bluetooth radios commonly used by mobile devices have a range up to 10 meters [12] and determining the distance between devices connected by Bluetooth is not trivial [14]. This would create additional security related concerns. As a result, while Bluetooth may be a possible solution to the problem stated, it is not the solution explored in this work.

Finally, using the front-facing camera, a user could scan a specific image to unlock the Glass. The solution presented in this paper, Glass OTP, relies on this method of input and will be discussed in detail in the next section.

3 Glass OTP

Glass OTP is a new and novel authentication scheme for the Google Glass device, which is designed to be both secure and convenient. The work presented here includes the development of two applications to support this authentication scheme: the "Glass OTP" lock screen for Glass devices and the "Glass OTP Companion" application for Android smartphones.

Glass OTP relies on a private key which is generated by the Glass OTP Companion application and shared with the Glass device. Using this private key, a time based OTP is generated by the Glass OTP Companion application and embedded in a QR code. The Glass OTP lock screen then uses the Glass camera to scan this QR code and verify the OTP before unlocking the device. Code from the ZXing project [6] was used to handle QR generation and QR scanning in the two Glass OTP applications.

3.1 Private Key and OTP Generation

The Glass OTP authentication scheme relies on the HMAC-SHA-256 algorithm to generate OTPs from random symmetric 256-bit keys. The key itself is generated by a smartphone using the Glass OTP Companion application. This application was developed to generate symmetric keys using the built in javax.crypto.KeyGenerator class [5]. Using this function, a 256-bit key is created and stored locally on the smartphone device in Android's "MODE_PRIVATE" [4]. Security of this storage method assumes the Android smartphone is not "rooted" or compromised in any way. As for the security of the algorithm chosen; as of the time of this paper's writing, SHA-256 has been successfully attacked up to 52 rounds out of 64 and still remains secure [10].

Glass OTP uses a time-based scheme following suggestions in IETF RFC 6238 [11] to generate passwords. The hashing algorithm used in Glass OTP, HMAC-SHA-256, is one of two algorithms suggested for implementation of TOTP schemes by the IETF. Glass OTP uses a 30 second time step based on the local system's current UNIX time. The OTP is obtained by first generating a hash of the current time step using HMAC-SHA-256 initialized with the shared private key. The resulting hash is then converted from the resultant byte array to a binary integer and modulo to the desired OTP length (6 digits in the case of Glass OTP). This implementation is used by both applications involved in the Glass OTP scheme to generate and verify OTPs.

It is realized that this implementation is not without possible vulnerabilities. Relying on local system UNIX time may allow for a replay attack using old OTPs and maliciously modified local time. Obtaining time from verified NTP servers may be one solution. Using local storage in "MODE_PRIVATE" [4] is also not foolproof. More secure means of storage include using the built-in "Android Key Store" or hardware-backed "KeyChain" credential storage, both of which were introduced with Android 4.3 (API 18) [15, 16]. The risks mentioned here were deemed acceptable for GlassOTP in its current stage of development and may be mitigated in the future.

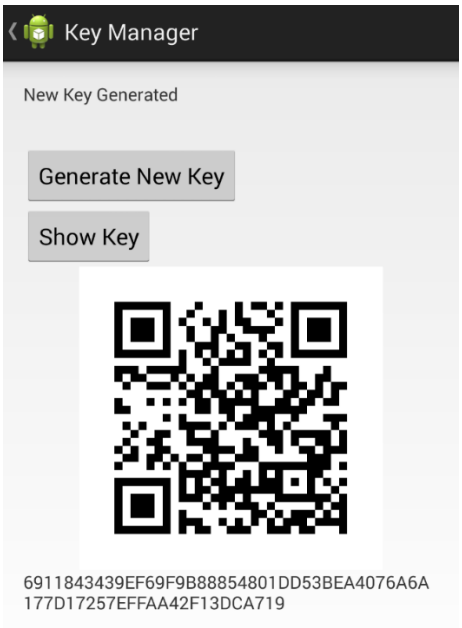

Fig. 1. The key management screen of Glass OTP Companion, where a private key is generated and displayed for scanning.

Fig. 2. The main/front screen of Glass OTP Companion where an OTP is generated and displayed for scanning.

3.2 Initial Setup Procedure

The setup procedure for Glass OTP is designed to be quick and requires minimal effort from the user.First, a secret key must be generated by the Glass OTP Companion application on a smartphone. The companion application can then embed the secret key in a QR code using ZXing libraries and display the code for a Glass device running Glass OTP to scan as seen in Fig. 1. Upon scanning the QR code, the Glass OTP lock screen will store the secret key locally and use it to verify scanned OTPs in the future. At this point, setup is complete.

3.3 Unlocking the Glass

Unlocking Glass using Glass OTP is a process designed to be quick and natural, requiring only a glance at one's smartphone. When a user tries to unlock their Glass, Glass OTP will prompt them to scan an OTP QR code. At this point, the user must launch the Glass OTP Companion application on their smartphone. When the companion application launches, it will automatically generate and display an OTP QR code as seen in Fig. 2. Scanning this OTP QR code with the Glass OTP lock screen will unlock the device.

4 Security and Usability Analysis

In a recent paper Bonneau et al. [9] analyzed existing alternatives to traditional passwords. Their work focused on authentication schemes implemented on traditional computing devices (e.g. a PC with keyboard and mouse). As a result, direct comparisons of Glass authentication schemes to analysis done in that research would not be useful. However, Bonneau et. al [9] do outline very specific and detailed criteria upon which to judge an authentication scheme. Below we analyze the proposed Glass OTP authentication scheme against these criteria.

The criteria are separated into three categories: usability, deployability and security. For each criteria, an authentication scheme either "offers the benefit," "almost offers the benefit," or "does not offer the benefit." And for each criteria, an authentication scheme is also evaluated to be "worse than passwords," "better than passwords," or "no change." [9] In our work, we will be comparing Glass OTP against gesture pattern authentication schemes where applicable.

Figure 3 summarizes the analysis against the given criteria and is followed by further explanation of each criterion.

Usability:

- Memorywise-Effortless: The Glass OTP requires nothing to remember. The user must remember to carry his or her phone, but Glass is currently designed as a companion device to smartphones. It can be reasonably assumed the user will carry their smartphone for other reasons as well. This solution is better than pattern locks, where users must memorize a specific sequence of gestures.

- Scalable-for-Users: Each user of Glass OTP is independent from other users and there is no reliance on a support infrastructure, which therefore makes it scalable.

- Nothing-to-Carry: User is required to carry a smartphone to use Glass OTP companion app. Pattern locks have an advantage here, as they do offer this benefit. However, since a large percentage of the population carries a smartphone[20], this does not change the usage model for these users.

- Physically-Effortless: Both OTP and smart phone require some physical effort on the side of the user. For OTP, the user is required to hold a smartphone in front of Glass for scanning, while for pattern locks, physical effort is required to perform gestures correctly. Both lock screens can be set to lock the device only when removed from a user's head [17].

	Glass OTP	Pattern Lock
Usability		
Memorywise-Effortless	●	
Scalable-for-Users	●	●
Nothing-to-Carry		●
Physically-Effortless		
Easy-to-Learn	●	●
Efficient-to-Use		●
Infrequent-Errors	●	
Easy-Recovery-from-Loss	●	●
Deployability		
Accessible		
Negligible-Cost-per-User	●	●
Server-Compatible	N/A	N/A
Browser-Compatible	N/A	N/A
Mature		
Non-Proprietary	●	
Security		
Resilient-to-Physical-Observation	●	
Resilient-to-Targeted-Impersonation	●	
Resilient-to-Throttled-Guessing	●	●
Resilient-to-Unthrottled-Guessing	●	●
Resilient-to-Internal-Observation	●	
Resilient-to-Leaks-from-Other-Verifiers	●	●
Resilient-to-Phishing	●	
Resilient-to-Theft	●	
No-Trusted-Third-Party	●	
Requiring-Explicit-Consent	●	●
Unlinkable	●	●

● = offers the benefit; no circle = does not offer the benefit; N/A = criteria not applicable

Fig. 3. Similar to the table used to compare criteria in "The Quest to Replace Passwords" (Microsoft [9]). This table compares Glass pattern locks and Glass OTP with the given criteria.

Usability (cont):

- Easy-to-Learn: Glass OTP has an easy-to-follow setup procedure and is designed to be simple for the user. This is similar to pattern locks, which also require learning as they are a new authentication scheme.

- Efficient-to-Use: Initial setup is required to exchange and share a private key for the Glass OTP. After initial one-time setup, pointing the Glass at a smartphone to scan a Glass OTP QR code is easy. Some Inconvenience/inefficiency may exist in having to launch the Glass OTP Companion application on the smartphone.

- Infrequent-Errors: Glass camera is able to accurately scan the Glass OTP QR code with ease. There are no possible user input errors to account for. This is better than pattern locks, where users must enter a sequence of gestures perfectly or try again.

- Easy-Recovery-from-Loss: A user is able to save their generated private key, either as text or as a screenshot of the private key QR code. Recovery involves scanning this code with a new smartphone running the Glass OTP Companion app. This is similar to the official Glass lock screen's recovery feature [8].

Deployability:

- Accessible: Glass OTP requires more physical effort due to necessary handling of the smartphone. Although the effort may be negligible, it must still be considered in this analysis. Pattern locks are similar as they require physical effort to lift one's arm to the proper position and perform the correct gestures.

- Negligible-Cost-per-User: Glass is designed as a companion device to smart phones. It is assumed that the Glass user will have a smart phone capable of running the Glass OTP companion app.

- Server-Compatible: Glass OTP functions are entirely client-side, with no requirement for web-hosted resources. Pattern locks are similar in this criteria.

- Browser-Compatible: Glass OTP isn't a meant as an everyday password replacement for multiple scenarios.

- Mature: This protocol introduces the first Glass security application that utilizes OTPs. Patterns locks for the Glass are also fairly immature, having existed only as early as April 2013 [7].

- Non-Proprietary: Glass OTP will be open source once released and would be non-proprietary. Google's pattern lock is proprietary [8], Bulletproof pattern lock is open source [7].

Security:

- Resilient-to-Physical-Observation: Any observed OTP QR code is time sensitive and useless after a short period of time, and is therefore resilient to physical observations. This offers better solution than pattern locks, where a user's gestures are clearly visible to nearby observers.

- Resilient-to-Targeted-Impersonation: Users decide how they want to keep their own private keys securely backed up. Better than pattern lock, as Google's pattern lock can be reset if an attacker obtains access to a victim's Google account [8].

- Resilient-to-Throttled-Guessing: It is infeasible to try all possible number combinations for the OTP before the current valid combination would expire, which renders it resilient to throttled guessing. Google's pattern lock has a lockout threshold [8]. Bulletproof has no limit on pattern length [7].

- Resilient-to-Unthrottled-Guessing: It is infeasible to try all possible number combinations before the current valid combination would expire. Google's pattern lock has a lockout threshold [8]. Bulletproof has no limit on pattern length [7].

- Resilient-to-Internal-Observation: An intercepted OTP would expire before it is useful. It is infeasible that an attacker could intercept an OTP, steal the user's Glass and authenticate before the OTP expires. This assumes the user does not store private key in an insecure manner. I.e. user does not store private key on Glass or smartphone in unencrypted form. The mechanism is better than pattern locks, where if a user's touchpad input is intercepted; the pattern of gestures for unlocking the device can be obtained and replayed.

- Resilient-to-Leaks-from-Other-Verifiers: All verification is done locally by Glass OTP, which renders it resilient to such leaks. Pattern locks also perform verification locally.

- Resilient-to-Phishing: All verification is done locally by Glass OTP. This is better than pattern locks, where a successful phishing attack on a victim's Google account would grant access to the Glass device if using pattern lock [8].

- Resilient-to-Theft: Compromise of Glass OTP requires theft of Glass device and paired smartphone. Resilience to theft depends on the security of the user's smartphone. This is better than pattern lock that does not rely on a separate physical device for authentication (only requires theft of Glass and pattern knowledge).

- No-Trusted-Third-Party: Glass OTP only allows a user to keep his own private key locally. All verification is done locally by Glass OTP. This is better than pattern lock which relies on Google services for pattern recovery.

- Requiring-Explicit-Consent: Glass OTP requires a user to purposefully use and position their phone in front of Glass.

- Unlinkable: All verification is done locally by Glass OTP. Same as pattern locks, which also verify locally.

Glass Camera Accuracy and Reliability. Another concern regarding the Glass OTP authentication scheme is the reliance on the Glass's front facing camera. Questions have been raised about if the camera can quickly and accurately scan QR codes displayed by the smartphone in various conditions. The distance a smartphone must be from the Glass camera is also something to be considered. Tests performed until the time of this paper's writing has yielded good results, however these tests were not done under properly controlled environments so official results were not recorded. The following results are recollections of experience from hands-on testing in everyday real world environments.

The Glass camera and ZXing code [6] (for QR code interpretation) is able to quickly and easily scan QR codes under most conditions. In normal light, such as outdoors in overcast conditions or indoors with regular lighting, QR codes are quickly and easy scanned. In low light or no light, the same was true since the QR code is being displayed by a smartphone screen which is backlit. In these cases, the smartphone displaying a QR code could be held away as far as one to two feet (almost half an average arm's length) away from Glass and still be successfully scanned

The only issues arose during testing in bright sunlight. In some cases, direct sunlight was able to overpower the backlighting of smartphone screens and make scanning the displayed QR code difficult to impossible. Indirect sunlight also posed an issue with smartphones which had insignificant maximum screen brightness. On newer phones, such as the HTC One M8 used for testing, max brightness was enough to display a clear QR code for the Glass to scan in bright but indirect sunlight. In the worst cases, the QR code on a smartphone could still be scanned if the user shaded the phone with his or her hand; bringing the phone closer to the camera also improved chances of a successful scan.

Key Backup and Recovery. A major concern with any kind of authentication scheme, especially ones which rely on a second physical token, is how to back-up and restore a key. With Glass OTP, the solution is fairly simple but places more responsibility on the user. Currently, a user can have the Glass OTP Companion application display the saved private key as both a string and QR code at any time. The easiest way to back up the private key would be to take a screenshot of the QR code representation and store it somewhere safe. If the smartphone used to unlock a specific Glass device is ever lost, a user just has to scan the private key QR code with the Glass OTP Companion application installed on a new smartphone.

5 Conclusions and Future Work

Wearable devices are improving and becoming increasingly popular as time goes on in our electronically connected society. Some people have a desire for as much information as possible, as fast as possible, and as accessible as possible. Wearables promise to fulfill such desires, but in doing so come with large privacy and security concerns.

Google Glass is such a device; it aims to keep a user connected at all times, containing and providing data relevant, specific, and sometimes private to the user. Unfortunately, as a result of its form factor, passwords as we know it cannot be used to secure the device. There is no keyboard to input long, random, and secure passwords.

Existing solutions aim to provide an alternative relying on gestures on the touchpad. But these gesture combinations are out in the open, for anyone nearby to see and worse, record.

This paper presents Glass OTP, an authentication scheme which finds a balance security and convenience not seen in existing authentication schemes. Glass OTP has offers resiliency to numerous security risks, similar to traditional One-Time-Password password schemes [9]. However, Glass OTP overcomes the inconvenience associated with One-Time-Passwords by using the Glass camera, sparing the user from having to manually input the password before it becomes invalid.

Future solutions could be introduced alongside hardware improvements by Google. For example, adding a biometric security mechanism similar to Apple's Touch ID [18] could prove to be a good solution compared to the alternatives examined in this paper. With the rising affordability of fingerprint readers, and as new applications -- such as the Paypal app[19] -- begin accepting fingerprints for authentication, integrating fingerprint readers into Google Glass may offer a promising authentication solution. However, this requires changes to the hardware that are not currently available on the system.

Future work should include a user study that examines the security and the usability of the OTP mechanism vs. the Pattern Lock mechanism. It should compare the actual time and the perceived time and convenience of using each approach. As users previously expressed significant concerns about privacy issues posed by Google Glass [21], the effect of adding a locking mechanism should be explored, and whether it will raise potential users' willingness to accept the system.

Acknowledgments. This work was supported in part by the NSF (under grant 1228842). The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of any of the sponsors.

References

1. Fitbit, Inc. Fitbit Official Site. http://www.fitbit.com. Accessed 2 October 2014
2. Google. Google Glass. http://www.google.com/glass. Accessed 20 September 2014
3. Google. Google Developers Voice Input. https://developers.google.com/glass/develop/gdk/voice. Accessed 8 October 2014
4. Google. Context Android Developers. http://developer.android.com/reference/android/content/Context.html. Accessed 1 October 2014
5. Oracle. KeyGenerator (Java Platform SE 7). http://docs.oracle.com/javase/7/docs/api/javax/crypto/KeyGenerator.html Accessed 1 October 2014
6. Github, Inc. zxing/Zxing. https://github.com/zxing/zxing. Accessed 12 October 2014
7. Github, Inc. kaze0/Bulletproof. https://github.com/kaze0/bulletproof. Accessed 10 October 2014
8. Google. Screen Lock – Google Glass Help. https://support.google.com/glass/answer/4389349?hl=en. Accessed 8 October 2014
9. Bonneau J, Herley C, van Oorschot P.C, Stajano F. The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes. Microsoft. http://research.microsoft.com/pubs/161585/QuestToReplacePasswords.pdf.
10. Li J, Isobe T, Shibutani K. Converting MITM Preimage Attack into Pseudo Collision Attack: Application to SHA-2 (2012). Sony China Research Laboratory. Sony Corporation. http://fse2012.inria.fr/SLIDES/67.pdf.
11. M'Raihi D, Machani S, Pei M, Rydell J. TOTP: Time-Based One-Time Password Algorithm (2011). Internet Engineering Task Force. https://tools.ietf.org/html/rfc6238.
12. Bluetooth SIG, Inc. Basics | Bluetooth Technology Website. http://www.bluetooth.com/Pages/Basics.aspx. Accessed 10 December 2014.
13. Google. Bluetooth Low Energy | Android Developers. https://developer.android.com/guide/topics/connectivity/bluetooth-le.html. Accessed 12 December 2014.
14. Ghose A, Bhaumik C, Chakravarty T. BlueEye – A System for Proximity Detection Using Bluetooth on Mobile Phones. UbiComp. http://www.ubicomp.org/ubicomp2013/adjunct/adjunct/p1135.pdf.
15. Google. Android 4.3 APIs | Android Developers. https://developer.android.com/about/versions/android-4.3.html. Accessed 16 December 2014.
16. Elenkov N. Jelly Bean hardware-backed credential storage. http://nelenkov.blogspot.com/2012/07/jelly-bean-hardware-backed-credential.html. Accessed 16 December 2014.
17. Google. On-Head Detection – Google Glass Help. https://support.google.com/glass/answer/3079857. Accessed 20 December 2014.
18. Apple. Apple – iPhone 6 – Touch ID. https://www.apple.com/iphone-6/touch-id/. Accessed 20 December 2014
19. FinExtra. PayPal adds fingerprint authentication to more Samsung devices. http://www.finextra.com/news/announcement.aspx?pressreleaseid=56577&topic=retail, 2014.
20. PewReseach Internet Project, Mobile Technology Fact Sheet, http://www.pewinternet.org/fact-sheets/mobile-technology-fact-sheet/

21. C|net, 72 percent say no to Google Glass because of privacy, http://www.cnet.com/news/72-percent-say-no-to-google-glass-because-of-privacy/
22. Design and Analysis of Shoulder Surfing Resistant PIN based Authentication Mechanisms on Google Glass, Dhruv Kumar Yadav, Beatrice Ionascu, Sai Vamsi Krishna Ongole, Aditi Roy and Nasir Memon, Wearable S&P 2015