On the Privacy, Security and Safety of Blood Pressure and Diabetes Apps

Konstantin Knorr[1,2], David Aspinall[1], and Maria Wolters[1]

[1] University of Edinburgh, UK,
david.aspinall@ed.ac.uk,maria.wolters@ed.ac.uk
[2] Trier University of Applied Sciences, Germany, knorr@hochschule-trier.de

Abstract. Mobile health (mHealth) apps are an ideal tool for monitoring and tracking long-term health conditions. In this paper, we examine whether mHealth apps succeed in ensuring the privacy, security, and safety of the health data entrusted to them. We investigate 154 apps from Android app stores using both automatic code and metadata analysis and a manual analysis of functionality and data leakage. Our study focuses on hypertension and diabetes, two common health conditions that require careful tracking of personal health data.
We find that many apps do not provide privacy policies or safe communications, are implemented in an insecure fashion, fail basic input validation tests and often have overall low code quality which suggests additional security and safety risks. We conclude with recommendations for App Stores, App developers, and end users.

1 Introduction

Mobile health (mHealth) applications cover all areas of health IT, from health information databases to personal electronic medical records. They are very popular—according to appbrain.com, in December 2014, Google Play had 21,457 apps in the medical category and 3% of these have been downloaded more than 50,000 times. mHealth apps for Android and iOS are currently largely unregulated [17]. Many of these apps are written by individuals or small companies who see a market niche, or by pharma and drugstore companies that seek to provide added value and collect information about their customer base [23].

Around 20% of medical apps cost money, on average US $9.78, making them the most expensive category. This suggests that users attach a high value to mHealth apps. But does this buy data protection?

Many mHealth apps handle highly sensitive data that require particular privacy and security precautions [13]. For example, insurers demand full disclosure of pre-existing conditions. If insurers find mHealth data that suggest an unreported condition, the applicant may be denied coverage or their policy may be downgraded.

In this paper, we examine how far a representative sample of 154 mHealth Android apps for two common long-term conditions, diabetes and hypertension, succeed in ensuring the privacy, security, and safety of the health data entrusted

to them. While previous work has considered relatively diverse samples of 40–50 mHealth apps at once [3, 6, 9, 10], we focus on a the management of long-term conditions which require users to regularly track key health indicators. Such conditions are an ideal mHealth use case. Well-designed apps allow users to enter data when and where they choose, to communicate with carers and health care professionals, and to discover trends and patterns in their own medical data.

Contributions. We introduce a novel method that takes into account the files contained in the APK (i.e., the downloaded package of the app), the dynamic behaviour of the installed app, and the app's privacy policy. We also consider input validation and source code quality. This comprehensive evaluation goes beyond previous studies, which mainly examined permissions and network traffic. In addition, our clear focus on long-term conditions allows an in-depth discussion of the specific privacy concerns.

The rest of the paper is structured as follows: Sect. 2 introduces mHealth apps and the categories we examine in detail; we explain the high-level functionality they provide, and discuss potential privacy threats. Sect. 3 describes the proposed methodology, while Sect. 4 gives the corresponding results and its discussion following the structure of the methodology. Suggestions for improving the current privacy issues of mHealth apps and future work conclude the paper in Sect. 5.

2 Hypertension and Diabetes

We focus on two long-term health conditions, hypertension and diabetes. People with hypertension suffer from chronically elevated blood pressure, which increases the risk of many serious illnesses including cardiovascular disease, stroke, and chronic kidney disease. Diabetes is a group of diseases that are characterised by elevated blood sugar levels. Controlling blood sugar is therefore the main aim of treatment.

Both conditions can be tracked using a simple numerical indicator, blood pressure (hypertension) and blood glucose level (diabetes), both are highly prevalent in the population[3], and for both conditions, self-monitoring is an important part of clinical management [14, 19].

The key function of a monitoring app is to capture a reading of the indicator measure at a given point in time. Fig. 1 shows a sample user interface for manual input of hypertension with a basic set of fields. Fields may not have a clearly defined use, such as "My Item" in our example, and others can contain free text, such as "Note".

Typical additional functionality includes providing reminders; data analysis and reporting; and backup, sharing, and export of recorded data. Some apps support several user profiles, while others include functionality for emergency texts and telephone calls. The main menu screen of *Diabetes Journal*

[3] Hypertension affects 25%–55% of the population depending on the country and the definition, while diabetes affects 8.5% of Europeans.

(`com.suderman.diabeteslog`, Fig. 2) shows how this functionality is typically implemented. "Averages" and "Charts" cover the data analysis and reporting function, the "Calendar" supports reminders, "Entries" leads to an entry screen for blood glucose values, and "Profiles" allows users to switch profiles.

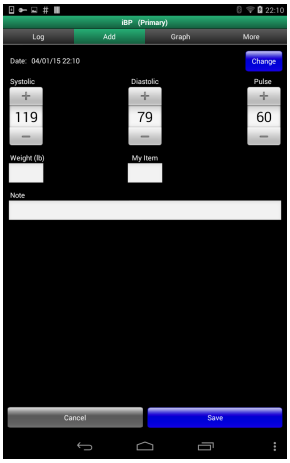

Fig. 1. Input in *iBP Blood Pressure*

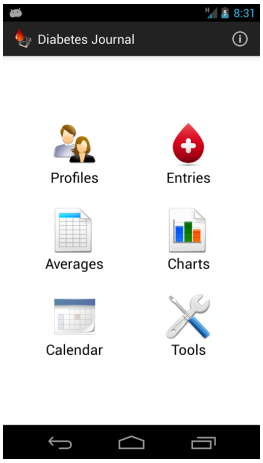

Fig. 2. Functions of *Diabetes Journal*

For both blood pressure and blood glucose levels, there are well documented clinical guidelines that govern their interpretation, such as those issued by the UK National Institute for Clinical Excellence (NICE). But integrating values collected using mHealth apps into clinical management is difficult. As long as most apps rely on manual data entry [7], data quality is questionable, especially given that, as we will see in Sect. 4, apps may fail to perform basic validity checks on the health data they receive from the user.

2.1 Privacy threats and relevant regulation

In the US, the Health Insurance Portability and Accountability Act (HIPAA) sets legal security and privacy standards for electronically transmitted health information; in the EU, there are diverse country-specific laws that need to be respected [5]. Devices that collect measurements which are used for monitoring, treatment, or diagnosis of health conditions are medical devices and these are regulated by the Food and Drug Administration in the US. The key medical device directive in the EU is Directive 93/42/EEC, which was updated in 2007.

Smartphone apps for monitoring diabetes and hypertension, in particular those that do not directly receive data from a validated blood pressure or blood glucose meter, fall into a grey area that is not adequately covered by current standards [11, 17], and finding a good balance between encouraging innovation and ensuring privacy is challenging [22].

Several frameworks have been proposed for classifying the threat that mHealth applications pose to the privacy of health information (e.g. [13]). Most of these focus on information that would make the patient identifiable, in particular ID or social security numbers. While many self-management apps do not collect those data, they do often store name, gender, and date of birth. Privacy threats also depend on the type of mHealth app [11], and on the condition that is being managed.

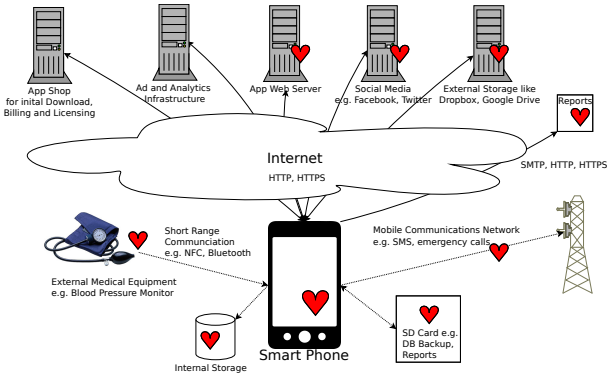

Fig. 3. An mHealth app in context. The heart indicates medical data.

2.2 Threat scenarios for mHealth apps in use

Fig. 3 shows a smart phone with mHealth apps installed. Medical data like blood pressure or glucose values can be input by the user or, less commonly, received from external medical devices via Bluetooth or NFC. The smartphone stores the app's data internally either in a database or the internal file structure. The SD card is used for database backups and restores, and to export reports or selected medical data. Mobile networks are used for emergency SMS and calls.

Five types of servers are potentially connected to the smartphone and app. The *App Shop* is used to initially download and possibly pay for the app. Subsequently, connections for licensing and billing e.g. for in-app purchases or upgrades of the app are maintained. Many of the apps display ads, which are provided by *Ad Servers*, most prominently *Admob* by Google. App developers also use *Analytics* such as Google Analytics to track user behaviour. Many apps allow to upload, backup or synchronize the data to *Social Media*, *Storage Services* or a dedicated external *App Web Server* provided by the app developer or cooperating third parties. This server may provide interfaces to other parties like insurance companies, doctors, or hospitals. Most apps allow sending *e-mails with medical data*. This is typically done via an Android e-mail "intent" which connects to a (implicitly trusted) email app on the phone. Overall, the complexity

in this picture emphasises the privacy challenges faced by a user trying to keep control over his/her medical data.

3 Methodology

We selected mHealth apps for diabetes and blood pressure which: (1) had an English or German user interface; (2) would run on a Google Nexus 7 test device with Android 4.4.2 installed; (3) had over 10,000 downloads (for free apps) or over 1,000 (for paid). Most apps came from the medical category and were specifically developed for diabetes and blood pressure monitoring. We ended up with 154 apps in the final test set (55% diabetes, 35% blood pressure, 10% both) that were installed on the Nexus 7. The most popular apps are `My Heart (com.szyk.myheart)` and `BP Watch (com.boxeelab.healthlete.bpwatch)` which each have between one and five million downloads.[4]

3.1 Our method

The investigation has four parts: (A) static analysis; (B) dynamic analysis; (C) web server security and (D) privacy policy inspection. Fig. 4 shows a picture of the method. The results were gathered in a database. Static analysis was applied to the full set of 154 apps; other analysis stages were applied to subsets.

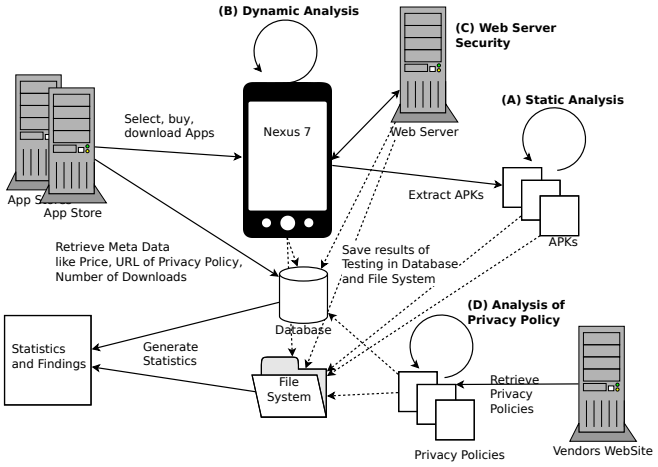

Fig. 4. Analysis method

[4] The full details of the apps analysed, along with a database of our results and details about the tools used can be found at the URL `http://tinyurl.com/mhealthapps`. A more comprehensive description of the method can be found in [12].

(A) Static analysis. The static analysis is based on the information contained in the APK file, including the manifest and the compiled code (in the file `classes.dex`). Results for the top 10 apps are shown in Table 1. We used MalloDroid [8] to identify *faulty SSL usage*, such as the failure to check certificate chains; this would allow man-in-the-middle attacks which enable an attacker to access all medical data transmitted. OpenSSL was used to extract certificate information and find certificates with poorly chosen cryptologic parameters [4].

We checked Android *content providers* and debugging flags using Drozer. The ContentProvider class allows sharing data between applications, with its own access control model. A careful implementation will prevent unauthorized access to sensitive data. When the *debug flag* for Android is set, the application can be debugged, even when running on a device in user mode, thereby possibly revealing medical information.

Addons Detector was used to identify and classify the add-on libraries used by the apps. Health apps were scanned for *malicious code*, security vulnerabilities and privacy failings using Snoopwall Privacy App, Clueful, AVG Antivirus Security, AVAST, McAfee, and the Recap vulnerability scanner.

We assessed *code quality* by using FindBugs to count the number of likely-bug patterns occurring in apps; high numbers indicate a likely poor code quality, which suggests possible unreliable behaviour, giving additional security and safety risks.

(B) Dynamic analysis. We analysed the 72 most frequently downloaded apps, including all those with web interfaces to external servers. We set up Facebook, Twitter, Dropbox, and Gmail accounts for a patient for whom some values were out of the physiologically normal range. The patient was 170 cm tall, weighed 99 kg, had dangerously high blood pressure (200 mmHg/120 mmHg), a physiologically improbable heart rate (333 bpm) and a blood glucose level of 111 mmol/L, which is lethal and indicates that the user confused mg/dL with mmol/L.

We generated Facebook, Twitter and Dropbox accounts for the patient and tested all available export routes for the data. We investigated whether abnormal and illegal inputs were accepted, and how exported data was stored or transmitted. Using the Android debugger command `adb pull` and the `adb logcat` command, we tested whether the backups or log data contained unencrypted medical data. We also established whether the app included a feature to erase all stored medical data, whether there was a privacy policy for the app, and whether the permissions required by the app were reasonable.

(C) Web server connection. For apps that can interface with a dedicated web server (n=20) using a user account for uploading data, we checked whether a sensible *password policy* was enforced on the web site and tested the *Web Server connection*. We recorded the URLs used for connections and noted if they used a secure transport (`https:`). We recorded traffic to see if passwords or medical data in textual or graphical form could be sniffed in clear text.

(D) Inspection of privacy policies. Only 19% of apps in (A) have a privacy policies. This is far lower than the proportion found by Sunyaev et al. [18] in their survey of the 600 most commonly used apps, where 30% of all apps had some form of privacy policy. Since storage of medical data on an external server further stresses privacy concerns, we limited the test to the same apps as in (C). We assessed privacy policies on *basic information* provided (URL to privacy policy, length in words, version, and country of origin), *completeness* (information about the OECD criteria accountability, security safeguards, openness, purpose, individual participation [2]), and *invasiveness* (possible use for other purposes, storage by third party, potential to be passed on to third parties) by asking basic questions and trying to find answers in the documents.

4 Results and discussion

Table 1. Analysis results for the top 20 downloaded apps. Columns: lack of a privacy policy, existence of MalloDroid errors, debug flag enabled, missing access protection of content provider, poor certificate parameters, usage of ad and analytics addons, more than 300 FindBugs errors, usage of more than 5 Android permissions, lack of input validation for input data, absence of a wipe feature.

Package	Downloads	Type	lacks pripol	bad SSL	debug set	c'provider	poor cert pars	analytics?	ads?	FindBugs >300	>5 perms	fail safety BP	fail safety GL	fail safety pulse	no wipe	**Score**
com.boxeelab.healthl	1000000	BP	×				×		×	×	×	×		×		7
com.szyk.myheart....	1000000	BP				×			×	×		×				4
com.ptashek.bplog...	500000	BP					×		×			×			×	4
com.gexperts.ontrack	500000	DIAB					×			×		×	×	×		5
com.fourtechnologies	100000	BP		×			×				×	×		×	×	6
com.orangekit.bpress	100000	BP	×						×	×					×	4
com.szyk.diabetes...	100000	DIAB					×	×	×	×						4
net.klier.blutdruck.	100000	BP		×			×					×	×	×	×	6
com.skyhealth.glucos	100000	DIAB			×		×	×							×	4
org.fruct.yar.bloodp	100000	BP	×				×	×	×	×	×	×				7
com.freshware.bloodp	100000	BP	×				×	×	×			×		×		6
com.jucdejeb.bloodpr	100000	BP	×				×		×	×		×		×	×	7
com.suderman.diabete	50000	DIAB					×	×	×	×		×	×	×	×	8
com.zlamanit.blood.p	50000	BP	×						×	×		×		×		5
com.freshware.dbees.	50000	DIAB	×				×					×		×	×	5
com.squaremed...typ1	50000	DIAB	×				×			×				×		4
com.squaremed...andr	50000	DIAB	×				×			×		×	×	×	×	7
com.mydiabetes.....	50000	DIAB	×				×	×	×	×			×			6
kr.co.openit.bpdiary	50000	BP					×			×	×	×			×	5
com.sidiary.app.....	50000	DIAB					×		×	×	×	×			×	6

Tables 1 and 2 give an overview of our results. In both Tables, × is used to indicate a problem, and the score is the number of problems found.

We found that paid apps tend to use fewer ads (even though some keep the ad libraries code) and offer more functionality (like e-mail or SD card export). Concerning the other tests, we could not find major differences. Especially comparing "twin apps" (free and paid version of same app) produced similar results.

Static analysis. The number of permissions used ranges from 0 (28 apps) to 17 with an average of 4.35. The most frequently requested Android permissions are `INTERNET` (126 times), followed by `WRITE_EXTERNAL_STORAGE` (117), and `ACCESS_NETWORK_STATE` (109). The permission `BLUETOOTH` is used 18 times, NFC 3 times. Of the 126 apps using the `INTERNET` permission, 15 do not correctly verify certificates or certificate paths, allowing for MITM attacks. Six apps had the debuggable flag set to "true" in their manifest, possibly allowing debug connections to inspect medical data. 17 apps use content providers that were accessible to other apps on the device, and four revealed medical data to all other apps on the device.

Apps are infested by ad and analytic addons on a large scale. The addons in the Advertising (74 apps, *Admob* being dominant) and Analytics (27 apps, *Google Analytics* dominant) category have been identified as a major privacy problems transferring device IDs and other data. Eight apps use Facebook or Twitter addons.

All apps used self-signed certificates, although using an acknowledged CA could establish additional trust. Certificates of 4 apps have a life time of ∼1,000 years, another 6 of ∼100 years. Most of them are valid for ∼30 years. The majority use SHA1, and only 17% the more secure SHA256. Only 40 certificates give more information than the name of the developer, thereby not establishing additional trust by adding more detailed information.

The Clueful tool gave a privacy score of 56 out of 100 for the apps on the test device and identified no high risk app and 57 moderate risk apps. The ranking is purely based on permissions. Snoopwall also just lists permissions for each app. Recap is a vulnerability scanner based on 1,800 CVE numbers. All its findings were OS related. We additionally checked if the apps in our test set in its current or earlier version are listed in the National Vulnerability Database `http://www.nvd.org` (NVD) and found no entries. AVG classified `com.stevenmz.BloodPressureDiary` as malware and `MMSL.BGGlucoDiary` as "Potentially Unwanted Program" without giving a detailed explanation[5]. Avast and McAfee did not find any malware, privacy or security issue.

Dynamic analysis. 43 of 50 blood pressure apps allow users to enter alarmingly high blood pressure values of 200 / 120 mmHg with the current time, some marking this (e.g. colouring the values read), some not. 20 of 50 blood glucose monitoring apps allow the meaningless reading of 111 mmol/L, and 24 of 46

[5] We did not find any suspicious network traffic caused by these two apps during the dynamic analysis

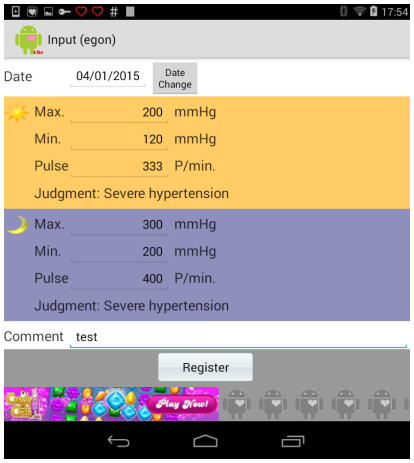

Fig. 5. *BloodPressure Record Lite* allows to enter lethal blood pressure values

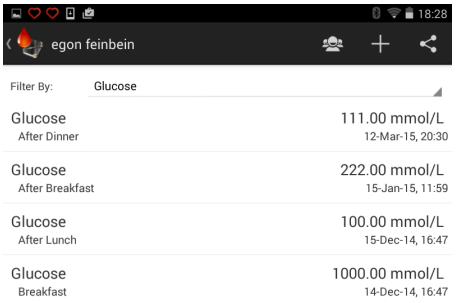

Fig. 6. *Diabetes Journal (Suderman)* allows to enter lethal blood sugar values

apps that monitor pulse allow a value of 333 bpm (s. Figs. 5 and 6). Some of the apps even allowed to input letters instead of numerical values. This raises serious safety concerns. Lacking input validation stands for poor code quality and is a precondition for attacks like SQL injection, XSS, and XSRF.

The majority of apps do not protect the medical data when stored in the internal storage. An attacker gaining physical access to the device can thereby access or manipulate these data. This attack can be impeded but not entirely prevented by using the phone's security mechanisms like PIN protection and device encryption. In contrast to [9, 10] we did not find any sensitive data in Android's main and event log. Only 4 apps allowed to protect the app with an extra password.

49 of 72 apps export medical data to SD card. All but one of these do not protect the data when doing so. The one exception is `MyLists`, which is not a specific health app. Thus, an attacker with physical access to the device can steal the SD card and gain thereby access to the medical data. This finding is surprising as some of the developer tools included in the apps like `PDFTron` would allow encryption. This also indicates *dead code* in the APKs. Only 32% of the apps provide a "wipe" feature to delete all medical data entered. Unfortunately, the SD card is often omitted from this wipe.

68% of the apps allowed to send e-mails. None of these allowed for sending encrypted messages or attachments. This could allow attackers with access to the network infrastructure (e.g. in WLANs) to sniff or even manipulate e-mails in transit. Besides export to 20 different web servers, we successfully exported data to Facebook (2 apps), Twitter (2), Google Drive (6), and Dropbox (8) without receiving a privacy warning by the app.

Several ad networks transmit the app's package name plus other device and app specific data in clear text which allows a passive eavesdropper to gather the information which device is using which app. This can often traced back to individual users thereby laying open that a user has diabetes or is tracking their blood pressure. We found this among others in HTTP traffic related to `doubleclick.net`, `googleapis.com`, `gstatic.com`, `applovin.com`. Similar findings (albeit not for mHealth) have been reported in [21].

Web applications. 35% of all analysed apps transmit medical data in textual or other forms (like pictures, charts, files) in clear text using HTTP. This is not limited to blood pressure or glucose values but also includes weight, BMI, and pulse (Fig. 7). The same percentage of passwords can be sniffed in cleartext. The password policies also leave much to be desired. One app allows empty passwords, 5 are fine with one character passwords, and only 3 ask for or generate passwords of 8 characters or more. Additionally, 17 of 20 apps accept password consisting just of numbers in a chronological order which allows easy brute force and dictionary attacks.

Table 2. Results of the privacy policy and web security analysis

Package	Downloads	HTTP?	password len <7	cleartext password	cleartext medical	accountability	safeguards	openness	purpose specified	Ind. Part. Princ. in Pripol	data for other purposes	3rd party storage	sell with company	share with 3rd parties
air.com.softbycloud.glico	1000				-	-	-	-	-	-	-	-	-	
androidhive.diabetes.....	1000	×		×	-	-	-	-	-	-	-	-	-	
com.diabetesstudio.client	500	×	×	×	×	×	×	×		×	-	×	-	
com.fourtechnologies.myne	100000	×	×		×	×	×	×	×	×	×	×		
com.freshware.dbees.....	50000		×		×	×			×	×	-	-	×	
com.glooko.logbook.....	5000				×	×	×	×	×	×	-	×	×	
com.kiwihealthcare.glubud	5000	×	×	×	×	-	-	-	-	-	-	-	-	-
com.leadingedgeapps.ibp..	5000	×	×	×	×	×	×	×	×		×	-	-	-
com.mysugr.android.compan	50000		×			×	×	×		×	×	-	-	
com.nabdacare.diabetes...	1000		×			-	-	-	-	-	-	-	-	-
com.oxygenhealthcom.lsmea	1000	×	×	×	×	×	×	×	×		-	-	×	×
com.sidiary.app.....	50000		×								-	-	-	×
com.socialdiabetes.androi	10000	×	×	×	×	×	×	×	×	×	×	-	-	
com.squaremed.diabetescon	50000		×			×	×	×	×	×	×	-	-	
com.squaremed.diabetesplu	50000		×			-	-	-	-	-	-	-	-	-
de.davidfroehlich.diabete	500					×	×				-	-	-	-
it.bytewave.glucowave....	500	×	×	×	×		×	×			×	-	-	-
net.klier.blutdruck.....	100000		×			×	×		×		-		-	
org.fruct.yar.bloodpressu	100000		×			-	-	-	-	-	-	-	-	-
org.fruct.yar.bloodpressu	1000		×			-	-	-	-	-	-	-	-	-

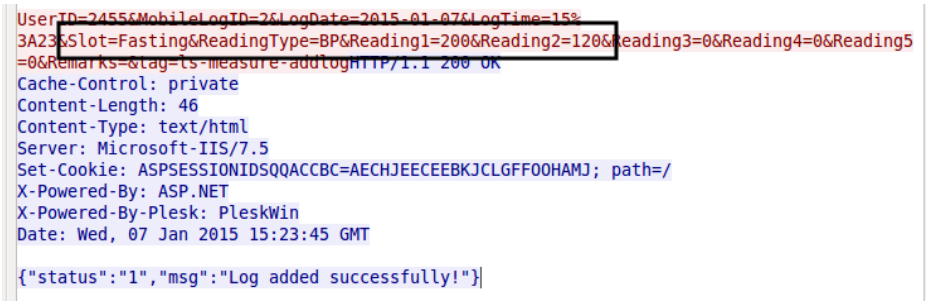

Fig. 7. Clear text blood pressure reading (`com.oxygenhealthcom.lsmeasure`)

Privacy policy. Only 19% of all 154 refer to a privacy policy in the app stores, but this number increases to 67% for apps which allow users to sync data with web servers. On average, the policies are 632 words long. Only 6 policies stem from 2014, the others being older, 4 give no date. mHealth is a global business with a trend towards USA and Germany. While the majority of the policies cover the OECD privacy principles of accountability, security safeguards, openness, and purpose at least partly, over half of the policies do not address the rights of the individual like the right of data deletion. None of the policies denies using the data internally e.g., for marketing or research. 9 policies explicitly say that they do so. All policies addressing mergers note that medical data may be transferred or even sold. 50% of the apps (that address this issue) say that medical data can be passed on to other 3rd parties (other than required by law).

5 Conclusions

Through our in-depth analysis, we found clear evidence of privacy, safety, and security concerns for the majority of the apps we analysed. Current health policy strongly encourages people to manage chronic conditions themselves, and mHealth is seen as a key tool for effective self management. But the apps people use should not leave them vulnerable to cyberattacks. While the consequences of such attacks may be relatively mild for the conditions we studied, hypertension and diabetes, they may be more severe for stigmatised conditions such as HIV+ or mental illness.

Some of the issues, such as the pervasive lack of encryption, indicate security is not a priority for developers. Reports, charts, and tables of medical data are often stored without any protection, giving thieves and eavesdroppers easy access. Another important threat is advertising. Of the 154 apps tested, 74 include advertisement addons. These addons often transmit the app's package name in clear text in the HTTP header which discloses the usage of this app (which is, per se, sensitive) to eavesdroppers. We also pointed out that current malware and privacy scanners fail to identify privacy issues in mHealth apps.

On the user interface side, we found that input data was often not validated or badly validated. This is a major concern when users want to share their self-curated data with health care professionals. The problems that arise from badly validated and designed data input forms have been studied extensively, and design guidelines have been formulated, but many app developers (and many medical device manufacturers) still fail to adhere to them [20].

App developers also rarely provide privacy policies. Although most users are unlikely to read such policies [15], we would still expect that privacy policies offer a reasonably complete summary of all major privacy issues for people who do read them. Most of the policies we analysed fell far short of this goal. Quote[6]: "Sinovo endeavours to use on the data minimally. The customer expressly agrees to the use of data in this context."

Recommendations. Due to the manifold concerns over the privacy and security of health data that users enter into mHealth apps in good faith, we suggest that *app shops* should mandate and enforce the existence of a privacy policy for apps that would like to be listed in the Health section. They could provide a template which systematically addresses the major principles and encourage automated security checking of apps with tools like MalloDroid or Drozer.

Such a step would automatically encourage *developers* to invest time and effort in ensuring users' privacy. Secure coding guidelines like [1] are a good start. Developers should leverage existing tools to encrypt all data stored both on the device and on external servers. Finally, developers should also ensure that the privacy policy is up to date, follows the OECD principles, and informs end users about use of their data and data protection.

Ad Networks, an important revenue source for free medical apps, are also a major source of privacy leaks. Here, many concerns could be addressed by mandating the usage of SSL to protect the HTTP traffic.

Looking at the *Android operating system*, the `INTERNET` permission seems too coarse. Currently it is not possible to differentiate whether a mHealth app wants to communicate with a health care provider or if an ad server is contacted. A possible solution could be the inclusion of firewall features in future Android versions, but this might conflict with Google's ad driven business model.

The recommendation to *end users* is to perform due diligence, including reading the description, privacy policy, and commenting on security issues, trying to take advantage of scores for privacy measures such as ours and those given by others [3, 6]. But the information currently available prior to installation is not enough for users to make these informed decisions. Therefore, a privacy aware user needs to use additional mechanisms like Android's phone encryption, ad block apps, or encryption apps. This is only feasible for technically savvy users.

Related work. Work closest to our contribution stems from [3, 6, 9, 10]. Njie [6] analyses 43 popular iOS and Android health and fitness apps based on a spreadsheet with the focus on network analysis and mentions the need to restrict future

[6] `http://www.sinovo.org/?id=144`

studies to a more specific category of apps. He et al. [10, 9] analyse 47 randomly selected iOS and Android mHealth apps in a series of two studies regarding Internet usage, logging, content provider, SD cards, and usage of cloud services and Bluetooth. Adhikari et al. [3] examine the 40 most popular iOS and Android mHealth apps by answering 9 privacy questions yielding a privacy score.

In contrast these studies, our work focuses (1) on apps for specific purposes, rather than selecting randomly from the broad category of mHealth apps and (2) only on Android apps. This gives us a more homogeneous population which can be closely tested and compared, in particular, it allowed us to check input validation for specific blood pressure and glucose reading ranges. Our testing also goes further than previous work: while permissions and network traffic have been considered, we additionally analyse the APKs and the underlying code and privacy policies, and source code quality.

Future work. Our methodology is sensitive enough to uncover many privacy and security concerns, and it can be easily extended to apps for conditions that are more stigmatised than diabetes and hypertension, such as mental health [16].

We propose to extend our technical work in three ways. First, we want to extend static analysis to more of the 27 topics in [1], integrating formal usability assessment, and developing automatic tools that might be used to screen new candidate apps for mHealth. Given the prevalence of web servers associated with apps, assessing web application security is an integral part of a full analysis; we'd like to extend this to consider vulnerability to attacks like SQL injection, XSS, and XSRF. Finally, apps that use communication links like Bluetooth and NFC to regulated measurement devices deserve to be examined in detail too, especially in a "wearable scenario".

In order to link our results to the eHealth field, we plan to investigate the motivations of users of mHealth apps and their attitude to the safety, security, and privacy problems we found. We also plan to investigate reasons why developers create insecure apps through a questionnaire study.

References

1. CERT secure coding standards for Android. Available online at `https://www.securecoding.cert.org`. Accessed 2014-12-28.
2. OECD guidelines on the protection of privacy and transborder flows of personal data. Available online at `http://www.oecd.org/internet/ieconomy/oecdguidelinesontheprotectionofprivacyandtransborderflowsofpersonaldata.htm`. Accessed 2014-12-29.
3. Rajindra Adhikari, Deborah Richards, and Karen Scott. Security and privacy issues related to the use of mobile health apps. ACIS, 2014.
4. Kevin Allix, Quentin Jerome, Tegawende F. Bissyande, Jacques Klein, Radu State, and Yves Le Traon. A Forensic Analysis of Android Malware: How is Malware Written and How It Could Be Detected? In *Proc. of the 38th COMPSAC*, pages 384–393. IEEE, 2014.
5. Sasikanth Avancha, Amit Baxi, and David Kotz. Privacy in mobile technology for personal healthcare. *ACM Computing Surveys*, 45(1):1–54, November 2012.

6. Craig Michael Lie Njie. Technical analysis of the data practices and privacy risks of 43 popular mobile health and fitness applications. Technical report, PrivacyRights Clearinghouse, August 2013.
7. Donna S Eng and Joyce M Lee. The promise and peril of mobile health applications for diabetes and endocrinology. *Pediatric diabetes*, 14(4):231–8, June 2013.
8. S. Fahl, M. Harbach, T. Muders, L. Baumgärtner, B. Freisleben, and M. Smith. Why eve and mallory love Android: An analysis of Android SSL (in) security. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 50–61. ACM, 2012.
9. Dongjing He. Security threats to Android apps. Master's thesis, University of Illinois at Urbana-Champaign, 2014.
10. Dongjing He, Muhammad Naveed, Carl A. Gunter, and Klara Nahrstedt. Security concerns in Android mHealth apps. In *Proceedings of the AMIA 2014*.
11. Anne Marie Helm and Daniel Georgatos. Privacy and mHealth: How Mobile Health 'Apps' Fit into a Privacy Framework Not Limited to HIPAA. *Syracuse Law Review*, 64, May 2014.
12. Konstantin Knorr and David Aspinall. Security Testing for Android mHealth Apps. In *Proceedings of the 6th international Workshop on Security Testing SECTEST, Graz, Austria, April 13, 2015*.
13. David Kotz. A threat taxonomy for mHealth privacy. In *3rd International Conference on Communication Systems and Networks, COMSNETS 2011*, 2011.
14. Alexander Labeit et al. Changes in the prevalence, treatment and control of hypertension in Germany? A clinical-epidemiological study of 50.000 primary care patients. *PloS one*, 7(12):e52229, January 2012.
15. Helen Nissenbaum. A Contextual Approach to Privacy Online. *Daedalus*, 140(4), 2011.
16. Carol Roeloffs, Cathy Sherbourne, Jürgen Unützer, Arlene Fink, Lingqi Tang, and Kenneth B Wells. Stigma and depression among primary care patients. *General hospital psychiatry*, 25(5):311–5.
17. Daniel F. Schulke. Regulatory arms race: Mobile-health applications and agency posturing, the. *BUL Rev.*, 93:1699, 2013.
18. Ali Sunyaev, Tobias Dehling, Patrick L Taylor, and Kenneth D Mandl. Availability and quality of mobile health app privacy policies. *Journal of the American Medical Informatics Association*, 2014.
19. T Tamayo, J Rosenbauer, S H Wild, A M W Spijkerman, C Baan, N G Forouhi, C Herder, and W Rathmann. Diabetes in Europe: an update. *Diabetes research and clinical practice*, 103(2):206–17, February 2014.
20. Harold Thimbleby. Improving safety in medical devices and systems. In *Proceedings IEEE International Conference on Healthcare Informatics*, 2013.
21. N. Vallina-Rodriguez, J. Shah, A. Finamore, Y. Grunenberger, H. Haddadi, K. Papagiannaki, and J. Crowcroft. Breaking for commercials: characterizing mobile advertising. In *Proceedings of the 2012 ACM conference on Internet measurement conference*, pages 343–356. ACM, 2012.
22. C Jason Wang and Delphine J Huang. The HIPAA conundrum in the era of mobile health and communications. *JAMA*, 310(11):1121–2, September 2013.
23. Maria Wolters. The minimal effective dose of reminder technology. In *CHI '14 Extended Abstracts*, 2014.