# Security Concerns in Android mHealth Apps

**Dongjing He, Muhammad Naveed, Carl A. Gunter, Klara Nahrstedt**
**Dept. of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL**

**Abstract**

*Mobile Health (mHealth) applications lie outside of regulatory protection such as HIPAA, which requires a baseline of privacy and security protections appropriate to sensitive medical data. However, mHealth apps, particularly those in the app stores for iOS and Android, are increasingly handling sensitive data for both professionals and patients. This paper presents a series of three studies of the mHealth apps in Google Play that show that mHealth apps make widespread use of unsecured Internet communications and third party servers. Both of these practices would be considered problematic under HIPAA, suggesting that increased use of mHealth apps could lead to less secure treatment of health data unless mHealth vendors make improvements in the way they communicate and store data.*

## 1. Introduction

The mHealth trend is evident: as of March 2013, Research2Guidance reported that there were about 97,000 mHealth apps across 62 app stores[1]. According to a report from MarketsandMarkets, the global mHealth market is predicted to grow from $6.21 billion in revenue in 2013 to $23.49 billion by 2018 at a compound annual growth rate (CAGR) of 30.5 percent over the five-year-period from 2013 to 2018. The mobile fitness and wellness market is expected to grow at a CAGR of 36.7 percent from 2013 to 2018[2]. This rising mHealth market threatens changes in the way significant amounts of health data will be managed, with a paradigm shift from mainframe systems located in the facilities of healthcare providers to apps on mobiles and storage in shared cloud services. This trend is paralleled by a new openness in which devices that were once only available in hospitals become widely available to individuals while flexible mHealth applications tempt clinicians away from the hospital-based systems they used in the past. This popular market will disruptively challenge traditional approaches by being cheap and accessible.

Security and privacy of health data could be significantly affected by this trend. Freed from the bonds of HIPAA, mHealth apps are free to handle data using lower assurances than those typically applied to HIPAA entities. However, the data they handle is often as sensitive as the data handled by HIPAA entities. Typical Google Play apps such as Self-help Anxiety Management, iCardio, Epocrates CME, and Clinical Advisor provide assistance with mental health concerns, activity monitoring, and information services that reveal user interests in particular symptoms or diseases. It is important to develop guidelines for the security and privacy of mHealth apps that suit a dynamic market while assuring that the growth of mHealth does not lead to a cavalier vendor attitude toward personal data. New security and privacy risks particular to mobile computing and communications technology abound in mHealth apps[3, 4]. The aspects of mHealth make it different from other health information systems: First, mHealth apps allow a much larger amount of data being collected from the patient, as mobile devices can collect data over extended periods of time. Second, a much broader range of health-related data is being collected, as many mHealth apps collect patient activities and lifestyle, not only physiological data, but also include physical activity, location tracking, eating habits and diet details, social interactions and so on. Third, the nature of communications technology and mobile computing exposes many new attack surfaces to the outside world.

The goal of this paper is to carry out a three-stage study of the security and privacy status of free mHealth apps offered on Google Play. In the first study, the top 160 free mHealth apps in Google Play are classified and examined to formulate a list of attack surfaces that need attention in this area. These are shown in Table 1. Then a random sample of 27 apps is selected from the top 1080 apps and analyzed with respect to these seven attack surfaces. Significant issues are found in three attack surfaces: *Internet, Logging,* and *Third Party Services*. Since our concern about *Logging* will be addressed to a significant degree by deployment of a new version of Android, we focus our attention on the other two: *Internet* and *Third Party Services*. A random sample of additional 22 apps is taken involving Internet communications. Examination confirms that many of these 22 apps display significant risks to security and privacy on these two attack surfaces. Our primary conclusions are that the mHealth apps in Google Play commonly send sensitive data in clear text and store it on third party servers whose confidentiality rules may not be as strong as they need to be for the type of data being stored.

**Table 1.** Description of attack surfaces.

| *Attack Surface* | *Description* |
|---|---|
| **Internet** | Sensitive information is sent over the Internet with insecure protocols, e.g. HTTP, misconfigured HTTPS, etc. |
| **Third Party** | Sensitive information is stored in third party servers |
| **Bluetooth** | Sensitive information collected by Bluetooth-enabled health devices can be sniffed or injected |
| **Logging** | Sensitive information is put into system logs where it is not secured |
| **SD Card Storage** | Sensitive information is stored as unencrypted files on SD card, publicly accessible by any other app |
| **Exported Components** | Android app components, intended to be private, are set as exported, making them accessible by other apps |
| **Side Channel** | Sensitive information can be inferred by a malicious app with side channels, e.g. network package size, sequence, timing, etc. |

The remainder of this paper is organized as follows. We first discuss background and related work, then describe our methods for the three experiments. The next three sections describe the three studies respectively. We end with discussion.

## 2. Background and Related Work

In recent years, we have seen an increased adoption of mobile health applications by patients and physicians as well as the general public[1, 2]. Mobile computing and communications technology bring about new security and privacy concerns[3, 4]. The main objective of our study is to systematically investigate the security and privacy risks in mHealth apps on the Android platform. To the best of our knowledge, our study is the first study for classifying Android mHealth apps and summarizing their security and privacy risks.

### 2.1. Related Work

Recently, researchers have been actively involved in mHealth research. Mosa et al.[5] review articles discussing the design, development and evaluation of mHealth apps and discuss the differences between apps for healthcare professionals, for medical and nursing students, and for patients. Martínez-Pérez et al.[6] review on commercial mHealth apps for the most prevalent health conditions in the Global Burden of Disease list provided by the World Health Organization. Kotz[4] develops a taxonomy of the privacy-related threats to mHealth. Through an extensive survey of the literature, Avancha, Baxi, and Kotz[3] develop a conceptual mHealth privacy framework and discuss the technologies that could support privacy-sensitive mHealth systems. Aarathi et al.[7] investigate patients' privacy concerns about sharing their health information collected from mHealth devices with their family, friends, third parties and the public. Our goal is to review commercial mHealth apps from Google Play in order to classify, analyze and demonstrate their security and privacy risks.

### 2.2. Android Operating System

Our work focuses on researching the security and privacy risks on Android platform. Android is an open-source platform supported by Google that has become the most common OS for mobile devices. A report by F-Secure[8] shows that Android attracts much more malware attacks than iOS, which is another popular mobile platform. There are many mHealth apps and solutions have been built for the Android platform[9, 10, 11, 12]. Android is based on Linux for mobile devices. It provides a rich application framework to allow developers to build apps written in Java. App components are the essential building blocks of an Android app. There are four different types of components: Activity, Service, Content Provider and Broadcast Receiver. Android uses *Intents* for inter-component communication. Intents are used to start an Activity, to start a Service, or to deliver a Broadcast message. An *Intent Filter* is an expression composed from action strings that specifies the types of Intents a component would like to receive. Android provides a *permission* mechanism to enforce restrictions of inter-component communication and access to system resources.

## 3. Methods

This paper investigates the security and privacy risks in Android mHealth apps. More specifically, we will investigate such threats in three studies:

**Study 1: What are the potential attack surfaces?**

We review 160 apps identified by selecting the top 80 free apps in Health & Fitness category and another top 80 free apps in Medical category from Google Play. In order to get a sense of the context of Android mHealth apps, we first divide the 160 apps into two groups with regard to their target users and classify them into eight categories according to their functionalities. To develop a list of attack surfaces that are most representative, we review research papers[13, 14, 15, 16, 17] and documents[8, 18], and we analyze the 160 mHealth apps to find evidence of threats. Based on this review, the following seven attack surfaces represent areas that need protection: *Internet, Third Party Services, Bluetooth, Logging, SD Card Storage, Exported Components,* and *Side Channels*.

**Study 2: How widespread is the threat?**

After identifying the potential attack surfaces, Study 2 takes a further step to learn how widespread these attack surfaces are. The top 1080 free apps are identified from the Medical and Health & Fitness categories on Google Play, 540 from each. By using a random number generator without replacement through random.org, 27 apps are selected for the dataset for Study 2. Of these apps, we analyze them one by one in detail with respect to the seven attack surfaces identified in Study 1. Three attack surfaces are identified as important ones: *Internet, Third Party Services* and *Logging*, because the majority of the 27 apps evidence issues with these attack surfaces. Figure 1 shows how we include and exclude apps for Study 2.

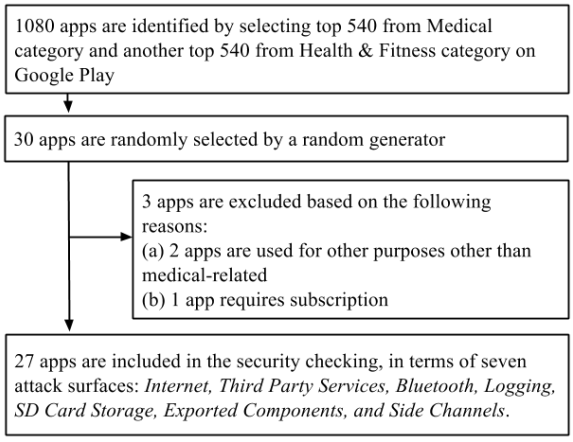


**Figure 1.** App selection flow graph for Study 2

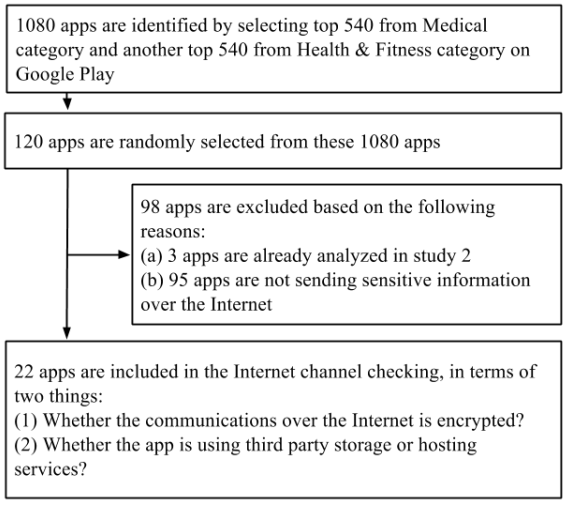


**Figure 2.** App selection flow graph for Study 3

**Study 3: How serious is the threat?**

Since security concerns in *Logging* will be addressed significantly by an Android version upgrade, we focus our attention on the other two attack surfaces, *Internet* and *Third Party Services*. The app selection process in Study 3 is similar to that of Study 2, but only apps that are sending sensitive information are selected. We randomly select 120 apps from the top 1080 free mHealth apps from Google Play. Then we use purposive sampling: the apps that have already been studied in Study 2 are excluded and the apps that are not sending sensitive information over the Internet are also excluded. In the end, 22 apps are included and analyzed in details to understand how serious the threat involving *Internet* communications is. Figure 2 shows how we include and exclude apps for Study 3.

**4. Results**

**4.1. Study 1: What are the Potential Attack Surfaces?**

To investigate the potential attack surfaces, we first want to understand the context of Android mHealth apps. By studying the 160 apps collected as described in the Section 3, we developed the classification system for Android mHealth apps shown in Table 2. We divide the top 160 free mHealth apps into two groups by their expected users. *Patient* apps are the ones mainly used by the individual whose health is being monitored. In most cases, the monitoring is done by the individual herself. *Healthcare Professional* apps are the ones mainly used by physicians, nurses, medical students, and other healthcare professionals to support their activities, which includes the monitoring

of patients. According to their functionalities, eight categories are used. Categories targeted at Patients include: *Lifestyle Management, Sensor-based Health Monitoring, Medical Contact, Medication and Disease Management,* and *Personal Health Record (PHR) Management*. Categories targeted at Healthcare Professionals include: *Medical References, Medical Training,* and *Clinical Communication*. A mHealth app may be useful for both Patients and Professionals (*e.g.* a pill identifier app can be used by patients to organize pills or by pharmacists to prevent errors in dispensing medications). Also, a mHealth app may belong to more than one category, since it may serve multiple functionalities (*e.g.* a fitness tracking app can monitor lifestyle data as well as manage PHR).

**Table 2.** Classification of popular free mHealth apps on Google Play.

| *Target users* | *Category* | *Functionality Examples* | *Modules Used* | *Number of Apps (%)* |
|---|---|---|---|---|
| **Patients** | Lifestyle Management | Count calories; track eating habits, exercise, sleep, period, pregnancy, and etc. | Accelerometer, Gyroscope, GPS, Network | 96 (60%) |
| | Medical Sensor-based Monitoring | Monitor health metrics such as: heart rate, blood pressure, blood glucose, insulin, cholesterol, and etc. | Externally connected health devices, Network | 15 (9.38%) |
| | Medical Contact | Contact registered nurses, doctors or hospitals | Network, Phone call, Email | 14 (8.75%) |
| | Medication and Disease Management | Manage prescription records; identify pills; shop medication online; look up symptoms; manage chronic diseases | Network | 27 (16.88%) |
| | PHR Management* | Manage and/or synchronize PHR with health services | Network | 75 (46.88%) |
| **Healthcare Professionals** | Medical References | Look up drug, disease and condition; anatomy tool; medical calculator; medical dictionary | Network | 26 (16.25%) |
| | Medical Training | Aid medical students studying medical theories | Network | 9 (5.63%) |
| | Clinical Communication | Emergency alert; photo sharing | GPS, Network | 2 (1.25%) |

* Here we define PHR management as patients syncing and managing user health information with an online health service provider.

Most of the applications in the categories are appropriate for our study but we exclude one app because it lacks a medical or healthcare purpose, and we exclude another app because its language is not English. Among the included 158 apps, we have 129 (81.65%) that are Patient-facing, 32 (20.25%) that are Professional-facing, and 3 (1.90%) drug identifier apps that are both. All the Patient-facing apps are from the Health & Fitness category and 41.03% of the apps from the Medical category are Professional-facing. In Table 2, the majority (60%) of the most popular Android mHealth apps are in the Life Management category. Nearly half (46.88%) of the apps manage and synchronize user health information to online service providers. The average rating score for the Patient-facing apps is 3.92, which is less than 4.18, the average score for the Professional-facing apps. However, the Patient-facing apps have almost 4 times more user installations, whose average is 502,263, than that of the Professional-facing apps, whose average is 139,125.

By studying previous literature[13, 14, 15, 16, 17] and online documents[8, 18], many different attack surfaces on Android apps have been identified. We study the 160 selected apps to have an understanding of what commercial mHealth apps are doing and whether risks exist in these attack surfaces. Real security issues are found within these Android mHealth apps, and the seven attack surfaces in Table 1 were identified as the most important ones. Here we use four specific examples in Android commercial mHealth apps to demonstrate the attack surfaces can lead to realistic and serious consequences.

***Case 1 (Unencrypted Internet):*** Many mHealth apps send unencrypted information over the Internet. For example, both Doctor Online[19] (patients can talk to doctors online) and Recipes by Ingredients[20] (patients can search recipes according to their illness or ingredients suitable for their diseases), send unencrypted sensitive information,

including the user's email and password, in clear text over the Internet. Figure 3 shows the network traffic from Doctor Online captured by WireShark that contains user's name, email and password.


**Figure 3.** Network traffic from Doctor Online containing sensitive information.

***Case 2 (Logging Sensitive Information):*** Many mHealth apps put sensitive user information into logs. For example, CVS/pharmacy[21], a popular app with millions of installations on Google Play, puts user login credentials and personal medical information in its log messages. Figure 4 shows the log messages with sensitive information from CVS/pharmacy. In the Example 1, CVS/pharmacy logs the prescription refill details from use inputs, including name, email address, store number, and Rx number. In the Example 2, CVS/pharmacy puts user login credentials in a debug log message. With this information a malicious party can view user profile and prescription history, which could support medical identity theft. A malicious party can even do pharmacy online shopping with users' stored credit card information.

**Example 1:** I/HttpDataClient(21039): https://native.usablenet.com/mt/ws/cvs.com/v2/refill?first=**tina**&last=**fey**&mail=**tina.health.droid%40gmail.com**&store=**24536**&orig=prod&rx1=**1524949**

**Example 2:** D/LOGIN (21039): https://native.usablenet.com/mt/ws/cvs.com/v2/login?username=**tina.health.droid%40gmail.com**&password=**password**&orig=prod

**Figure 4.** Log messages from CVS/pharmacy containing sensitive information.

Note that in both cases, the sensitive information is contained in the query strings of HTTPS URLs. Some developers may have a misconception that all HTTPS requests using GET or POST are sent over encrypted TCP connections so that sensitive information can be safely put into HTTPS URLs. However, even if sensitive information is not seen during transit, it remains visible in other places, such as mobile app logs, server logs, browser history and so on. Developers should avoid as far as possible including sensitive information in logs since it may be hard to know or control who is able to access the logs.

***Case 3 (Exported components):*** Several apps in our study have component exposure threats. For instance, Noom Weight Loss Coach[22], an app with more than 10 million installations, exposes its Content Providers to external apps, which means any app can access the exposed Content Providers without declaring any permission. After searching for "content://" paths in the manifest and decompiled source code, we get a list of content URIs defined in the app. By using Drozer[23], an automatic security analysis tool, we attempted to access sensitive information with each content URI. Figure 5 shows the ability to read user workout history stored in the app's Content Provider with the content URI "content://com.wsl.noom.exerciseinfo".

**dz> run app.provider.query content://com.wsl.noom.exerciseinfo**

| accessCode | key | timestamp | exerciseType | duration | caloriesBurnt | isManual | stepsCalories |
|---|---|---|---|---|---|---|---|
| WUX2G6EF | 0 | 1393052360350 | Walking | 50892 | 0.484559 | 0 | 0 |
| WUX2G6EF | 1 | 1393009200000 | Running | 1800000 | 262 | 1 | 175 |

**Figure 5.** Access Noom Weight Loss Coach's user workout history by using Drozer.

***Case 4 (Unencrypted SD card storage):*** Some sleep monitoring apps, such as SnoreClock[24] and Sleep Talk Recorder[25], record the sleep sounds of users and store them as unencrypted audio files on an external storage. For instance, the Sleep Talk Recorder explicitly stores sleep-recording unencrypted audio files on the SD card with name format YYYY-MM-DD-HH-MM-SS.wav. With read storage permission, a malicious app can read a user's sleep recordings; with internet permission, it can further send this information to remote servers. Another example is that Urgent Care[26] stores system logs in an unencrypted file on SD card, potentially leaking symptom lookup history.

### 4.2. Study 2: How Widespread is the Threat?

The complexity of the Android system has led to numerous potential attack surfaces that could be exploited by a malicious party to gain unauthorized access to sensitive data in mHealth apps and cause serious consequences. Analyzing these attack surfaces can help security specialists do security assessment and help mHealth users and developers understand and manage security risks. In Study 2, we analyze these attack surfaces with a new set of 27 random selected Android mHealth apps. The process of selecting these apps is described in Section 3.

***Internet:*** Android mHealth apps access the Internet for various purposes, including to transfer information to a remote server and to retrieve ad to display to users. The information transferred over the Internet to a remote server

includes sensitive health information and ideally all such communication with the remote server should be encrypted. The 27 randomly sampled apps are analyzed to study why they require the Internet access (i.e. to transfer information or to display ads). Furthermore, we analyzed if the encrypted communication is used in network transmission.

Any app can get access to the Internet with Andoid's INTERNET permission. To study if the apps are using Internet for displaying ads or transferring information to the remote server, we study the description of the apps and check the functionality of the apps by installing and using them on a Samsung Galaxy SII phone. As a result, 85.2% (23/27) of the apps have the permission to access the Internet. 70.4% (19/27) use the INTERNET permission to display ads, while 29.6% (8/27) of them use it to communicate user information over the Internet.

To study whether the communication with remote servers is encrypted we installed and ran each of the apps while capturing network traffic using the "Shark for Root" Android app, and used WireShark to see if the traffic is encrypted. The result shows 7.4% (2/27) of the apps allow the users to use the blog or social network associated with the app via the Internet but only one of the apps used encrypted communication. We found that 25.9% (7/27) transmit medical information to the remote server, 57.1% (4/7) use encrypted communication, and 42.9% (3/7) use unencrypted communication to transfer the sensitive health related information.

We analyzed if the three apps sending unencrypted data over the Internet are actually sending sensitive information. The first app searches for nearby pharmacies, doctors, etc. The second app tracks exercise workouts, and the third (Doctor Online from Spain) facilitates finding and talking to doctors online. Doctor Online sends email, username and even password unencrypted over the Internet.

***Third Party Services***: Android apps use storage and hosting services such as Amazon instead of maintaining their own infrastructure. This is an economical as well as scalable solution for mobile apps. But storing sensitive health information on these third party services can have serious implications even for large and widely-trusted services like Amazon. We study if these seven apps communicating with remote servers are hosted on the cloud or on-premises servers owned by the app vendors. To this end, we analyze the IP addresses of these apps in the communications with their respective servers. IP addresses have a publicly available record of whom it belongs to and we use this to find out where the traffic is going. We found that 85.7% (6/7) apps are hosted on third party servers. Three of them are hosted on Amazon and rest on other hosting services. We were not able to tell if data on the remote third party servers is stored in encrypted fashion such that the hosting companies do not have access to this data. However, the four apps mentioned in the previous Internet section are using encryption for the communication only.

***Bluetooth:*** Many mHealth sensing apps primarily use Bluetooth to collect data from health sensors to mobile devices. One app (3.7%) of the 27 apps in our dataset connects to a Bluetooth health device to collect personal health information. Supporting Bluetooth devices is more common among the 160 most popular Android mHealth apps, where 15/160 (9.5%) provide Bluetooth connectivity to collect health data. 12 of the 15 apps declare and use both BLUETOOTH and BLUETOOTH_ADMIN permissions, so that they can use Bluetooth to connect and collect data from external health sensors, while the remaining three of them collect health data via the Internet or by connecting with other apps. The apps collect various types of health information, including heart rate, respiration, pulse oximetry, electrocardiogram (ECG), blood pressure, body weight, body temperature, quality of sleep, exercise activities. Apparently, Bluetooth is a major communication technology for sensor-based health monitoring in Android mHealth apps. Naveed et al.[14] present a problem of external-device misbonding (DMB) for Bluetooth-enabled Android devices and health sensors. They show how a malicious app can stealthily collect user data from an Android device or spoof a device and inject fake data into the original device's app. One app of the 27 apps connects to external health sensors and uses default PIN code 0000, which makes it vulnerable to the DMB attack. To defend against the Bluetooth-based threats on mHealth apps, Naveed et al.[14] propose an OS-level protection, which generates secure bonding policies between a device and its official app and enforces these rules when establishing and terminating Bluetooth connections.

***Logging:*** the Android logging system enables developers to collect and view debugging output for apps. The logging facility allows a system-wide logging, including both application information and system events. If an app is granted READ_LOGS permission, the app is allowed to read the low-level system log messages. With the READ_LOGS permission, a malicious app may be able to extract sensitive information from log messages. To find such logging vulnerabilities we used a tool called logcat from the Android Debug Bridge (ADB) shell to view system log messages.

In our dataset, 9 out of the 27 apps (33.3%) put sensitive information in log messages. Among the 9 aforementioned apps, two (22.2%) disclose GPS coordinates, three (33.3%) disclose Facebook friend information, and one (11.1%) divulges more sensitive data such as user sign up data, which includes name, location and profession of the user. Three (33.3%) apps leak disease and drug browsing history in the app logs. From the study on our dataset, it indicates a large number (33.3%) of the mHealth apps leak sensitive information in system logs that could support cause serious attacks such as medical identity theft.

***SD Card Storage:*** Each Android app gets a dedicated part of file system where it can write its private data. However, if an app writes files to an external storage, such as an SD card, the files are not guaranteed to be protected. With READ_EXTERNAL_STORAGE or WRITE_EXTERNAL_STORAGE permissions, any app can read or write files from an external storage. Before API level 19, the READ_EXTERNAL_STORAGE permission is not enforced and all apps still have access to read from an external storage.

In our dataset, 66.7% (18/27) of the apps declare the WRITE_EXTERNAL_STORAGE permission, which means they write data to external storage that can be read by any app with the READ_EXTERNAL_STORAGE permission. We used Dex2jar[27] to decompile the application package (apk) files for these 27 apps to get their Java source code. We searched this code for the "ExternalStorage" and "ExternalFiles" keywords in the source code to construct all possible paths for files stored on SD card. Then, we executed all possible operations with the studied apps and exhaustively went through the resulting directories to check their file contents. This search did not reveal evidence that any of the apps store sensitive information in external storage files.

***Exported Components:*** Android app developers can specify if a component (Activity, Service, Broadcast Receiver, or Content Provider) is public to external apps. A component can be declared as *exported*, or public, if its declaration sets the EXPORTED flag or includes at least one Intent Filter without permission protection. However, setting a private component improperly as *exported* enables a malicious app to send unwanted Intents to the component, which can cause security problems with broadcast injection, activity launch or service launch[17]. In addition, if the Content Provider is exported, a malicious app can read or write the exported Content Provider without declaring any particular permission. The Content Provider supports the basic "CRUD" (create, retrieve, update, and delete) functions and the data in a Content Provider is addressed via a "content URI". Knowing the "content URI" from an exported Content Provider, a malicious app can retrieve or modify the data according to the Content Provider's schema. An example we gave earlier in our discussion of Study 1 illustrates an unauthorized access to an exported Content Provider to read the app's sensitive information.

***Side Channels:*** All the attack surfaces discussed above are using explicit channels in Android system, where a malicious party has chance to directly read sensitive data from the attack surfaces. Besides the explicit channels, side channels can be exploited by a malicious party to infer sensitive information from apps, even if they are well-designed and implemented by their developers. Zhou et al.[13] find a correlation between network payload size, which is publicly accessible in Android system, and the disease condition a user selects on WebMD mobile[28]. With this correlation even an app with no permissions, a "zero-permission" malicious app, can monitor WebMD's network payload in the background, and map their monitoring results to the disease condition that a user searches on WebMD. Side channel information leakage has also been discovered from motion sensors, such as accelerometer and gyroscope[15, 16]. Fine-grained motion sensor monitoring can be used to infer keystrokes, such as 4-digit PIN codes, on touch screen smartphones with only soft keyboards.

To circumvent the network-payload-based side channel attack, Zhou et al.[13] present a mitigation mechanism which enforces limitations to accessing Android public resources (*e.g.* network payload size) by modifying the Android kernel. MHealth developers can pad blank information to network packets to ensure they are fixed-length, or develop offline strategies for downloading sensitive data. For the motion-sensor-based side channel attack, Adam *et al*.[15] propose disabling untrusted access to motion sensors whenever a trusted input function (*e.g.* password entry) is being performed.

### 4.3. Study 3: How Serious is the Threat?

Three vulnerabilities in Study 2 are revealed to be common and serious: sending sensitive information unencrypted over the Internet, storing it on third party services, and including it in logs. Since logging can be addressed by an Android version upgrade, we focus our investigation on the other two threats, Internet traffic and third party services.

As only seven apps in Study 2 are actually sending sensitive information over the Internet, we carried out Study 3 to understand the prevalence of these threats with a larger number of apps using the Internet. Another 120 apps are

randomly sampled from the 540 top Health and Fitness apps and another 540 top medical apps (1080 in total) from Google Play. These apps are then manually analyzed to rule out those not sending any sensitive information over the Internet. After the filtering, 22 apps are found to be sending sensitive data over the Internet (some apps requiring subscription are filtered out as well). To analyze their Internet traffic, we installed these 22 apps and captured their traffic using the same methods described in Study 2. The results reveal that 63.6% (14/22) of these apps are sending unencrypted data over the Internet and 81.8% (18/22) are using third party storage and hosting services such as Amazon's cloud services. One of our randomly selected apps (Fitbit) uses encryption over the Internet, but is also using third party storage and hosting services. The four apps that are using their own servers to store and host their apps are big companies such as Aetna, United Healthcare, Caring Bridge and US Dept. of Health and Human Services. We were not able to obtain ground truth about whether apps encrypt data when they store it with third parties, but one may conjecture that apps that do not encrypt data over the Internet probably also do not encrypt it on third party storage. Even though the data might be hosted in an isolated environment (e.g. on an isolated VM in the cloud), storing unencrypted data on third party storage makes the data vulnerable to insider attack, where the service provider is malicious.

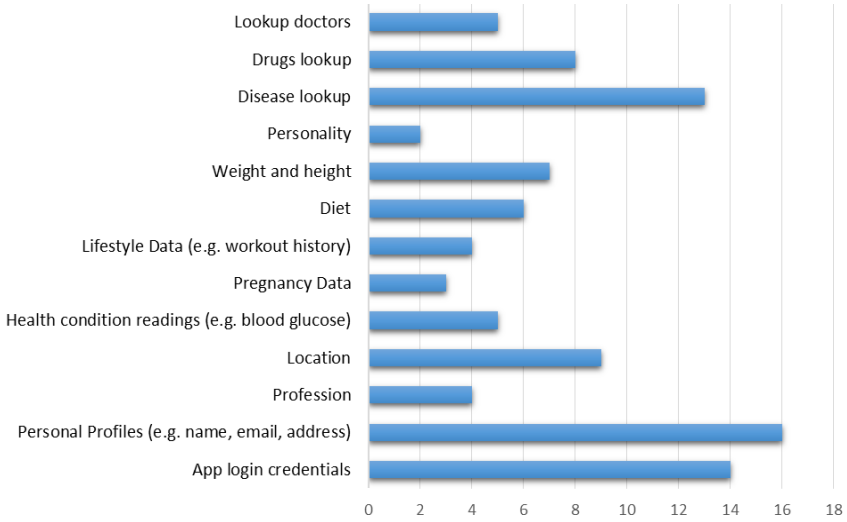


**Figure 6.** Sensitive information distribution in the 22 apps dataset for Study 3.

When used as intended, a variety of sensitive user data are collected, stored, and transmitted through these Android mHealth apps. Figure 6 shows the distribution of sensitive information in the 22 apps (x axis means the number of appearances of sensitive information in the 22 apps). Based on our study, the information includes at least personal profiles, health sensor data, lifestyle data, medical information browsing history, and third-party app data (*e.g.* Facebook account information). Depending on the type, sensitivity, and volume of mHealth data breaches, disclosure or tampering with these sensitive data may lead to serious consequences, such as profiling, medical identity theft, and healthcare decision-making errors. According to the information collected from World Privacy Forum[29], thefts have used stolen medical information for a resourceful collection of nefarious purposes. For example, a Colorado man whose Social Security number, name and address had been stolen received a bill for $44,000 he presumably owed to a hospital because his identity had been used by a thief to get medical services in his name. In another case, another identity thief in Missouri used the personal data of multiple victims to establish false driving licenses and was able to use them to obtain prescriptions in the victims' names at a regional health center.

## 5. Discussion

### 5.1. Summary of Findings

Our three-stage study raises many concerns and shows some serious problems with Android mHealth apps. The major issue is unencrypted communication over the Internet and use of thirty party hosting and storage services. Our study shows that a significant number of the apps in the top health related apps from the Google Play market have these issues. These issues need attention and are not easily fixable because they require extra effort and security expertise from developers and computational capabilities from platforms. Third party cloud and hosting services

provide a very economical solution for hosting app services and storing data and many app vendors may feel it is not economical to maintain their own servers or even encrypt stored data on the third-party services.

### 5.2. Compliance Recommendations

The increased use of mHealth results in greater risks to health-related information on mobile devices. Developers and healthcare service providers would be wise to make efforts to ensure that mHealth apps facilitate security compliance even if they are not legally required to do so (at the current time). Based on our study on the risks from mHealth apps, here are some important compliance recommendations: encryption is essential to secure personal data stored on mobile devices; when accessing web-based services, TLS/SSL should be deployed throughout the Internet transmission session; even though the network transmission session is protected and encrypted, using third party services to store users' sensitive data must be closely reviewed and users should be informed when it is happening; developer guidelines or training can be helpful in avoiding many of the common mistakes that are rooted from development with poor security practices; risk assessment provided by authorities can further minimize the security risks that may harm users. Experience with Happtique, which was forced to suspend app certifications after some of the apps it certified were found to have some of the problems above, provides evidence of strong incentives for better security and privacy practices[30]. A report from Symantec also raises questions about security risks in self-tracking devices and apps[31]. A good possible direction is for mHealth app developers to create a set of security and privacy guidelines that offer a baseline for protections.

### 5.3. Limitations of the Study

***Android version upgrade.*** Android is constantly making behavior changes in order to circumvent newly found threats. For example, to mitigate the logging information leakage problem, since Jelly Bean (Android 4.1), an app can only collect and view log messages originating from itself. However, on a rooted device (i.e., a device allows any app to run with administration permissions on Android)[32], a malicious app can, by executing a '*pm grant'* command, grant itself a READ_LOGS permission. This means it is still dangerous for an app to keep sensitive information in system logs. According to the Android platform distribution[33] collected in March, 2014, almost 40% of the overall Android devices are under the version of Jelly Bean. Due to a large number of Android device users and mHealth apps, it is lucrative for malicious parties to investigate ways to harvest sensitive personal healthcare information from mHealth apps.

***User agreements.*** We observed that many apps may ask users to share their private health information by providing privacy policy agreed by users themselves. In our study, most of the apps do make privacy policies available to users either via an URL link in the app or shown when the app is launched for the first time. How health data is managed and transmitted is generally out of control or visibility of the users, but the apps should at least encrypt all data in transit and at rest. We believe that understanding the privacy policies of mHealth apps is an interesting future research topic. Users should know what they are agreeing to in order to use the app and how their data can be used.

## 6. Conclusion

A study of Android mHealth apps reveals common shortcomings in security and privacy when using communications and storage. Steps should be made to encourage mHealth app vendors to assure encrypted network links for communications and the use of third party storage only when adequate security and privacy guarantees are obtained.


## Acknowledgements

We acknowledge the grant HHS-90TR0003/01 (SHARPS, sharps.org) from the Office of the National Coordinator for Health Information Technology at the Department of Health and Human Services (HHS) and NSF 13-30491 (THaW, thaw.org). The views in this paper represent opinions of the authors only.


## References


1. Aitken M, Gauntlett C. Patient apps for improved healthcare from novelty to mainstream. Parsippany (NJ): IMS Institute for Healthcare Informatics; 2013 Oct.
2. MarketsandMarkets. Mobile health apps & solutions market by connected devices (cardiac monitoring, diabetes management devices), health apps (exercise, weight loss, women's health, sleep and meditation), medical apps (medical reference) – global trends & forecast to 2018. 2013 Sep. Report No.: HIT 2104
3. Avancha S, Baxi A, Kotz D. Privacy in mobile technology for personal healthcare. ACM Computing Surveys (CSUR). 2012; 45 (1): 3.

4. Kotz, D. A threat taxonomy for mHealth privacy. 2011 Third International Conference on Communication Systems and Networks (COMSNETS); 2011 Jan 4-8; Bangalore.
5. Mosa A, Yoo I, Sheets L. A systematic review of healthcare applications for smartphones. BMC Medical Informatics and Decision Making; 2012 Jul 10; 12(7). BioMed Central.
6. Martínez-Pérez B, de la Torre-Díez I, López-Coronado M. Mobile health applications for the most prevalent conditions by the world health organization: review and analysis. J Med Internet Res 2013 Jun 14; 15(6):e120.
7. Prasad A, Sorber J, Stablein T, Anthony D, Kotz D. Understanding Sharing Preferences and Behavior for mHealth Devices. Proceedings of the 2012 ACM Workshop on Privacy in the Electronic Society (WPES). October 15; Raleigh, NC. New York, NY: ACM; 2012. p. 117-128.
8. F-Secure Labs. Mobile threat report. Helsinki, Finland; 2013 Jul-Sep. Report No. 2013 Q3. http://www.f-secure.com/static/doc/labs_global/Research/Mobile_Threat_Report_Q3_2013.pdf
9. Rajput Z, Mbugua S, Amadi D, Chepng'eno V, Saleem J, Anokwa et al. Evaluation of an Android-based mHealth system for population surveillance in developing countries. J Am Med Inform Assoc; Feb 24; 2012.
10. Altini M, Penders J, Roebbers H. An Android-based body area network gateway for mobile health applications. Proceedings in Wireless Health 2010; 2010 Oct -710-13; San Diego, CA. New York, NY: ACM; 2010. p. 188-189.
11. Wei R, Yang Z. Design and implementation of doctor-patient interaction system based on android. 2012 International Symposium on Information Technology in Medicine and Education (ITME); 2012 Aug 3-5; Hokodate, Hokkaido; 2: 580-583.
12. Gregoski M, Vertegel A, Treiber F. Photoplethysmograph (PPG) derived heart rate (HR) acquisition using an Android smart phone. Proceedings of the 2nd Conference on Wireless Health; 2011 Oct 10-13; San Diego, CA. New York, NY: ACM; 2011: 23.
13. Zhou X, Demetriou S, He D et al. Identity, location, disease and more: inferring your secrets from Android public resources. 20th ACM Conference on Computer and Communications Security (CCS); 2013 Nov 4-8; Berlin, Germany. New York, NY: ACM, 2013.
14. Naveed M, Zhou X, Demetriou S, Wang XF, Gunter CA. Inside job: understanding and mitigating the threats of external device mis-bonding on Android. Proceedings of the 21st Annual Network and Distributed System Security Symposium (NDSS); 2014 Feb 23-26; San Diego, CA. Reston, VA: The Internet Society, 2014.
15. Aviv A, Sapp B, Blaze M, Smith J. Practicality of accelerometer side channels on smartphones. Proceedings of the 28th Annual Computer Security Applications Conference (ACSAC). December 9-13; Orlando, FL. New York, NY: ACM; 2012. p. 41-50.
16. Cai L, Chen H. On the practicality of motion based keystroke inference attack. Proceedings of the 5th International Conference on Trust and Trustworthy Computing (TRUST). June 13-15; Vienna, Austria. Berlin, Heidelberg: Springer-Verlag; 2012. p. 273-290.
17. Chin E, Felt AF, Greenwood K, Wagner D. Analyzing inter-application communication in Android. Proceedings of the 9th international conference on Mobile systems, applications. June 28-July 1. New York, NY: ACM; 2011. p. 239-252.
18. Seven ways to hang yourself with Google Android. http://www.cs.berkeley.edu/~emc/slides/SevenWaysToHangYourselfWithGoogleAndroid.pdf
19. Doctor Online. https://play.google.com/store/apps/details?id=com.airpersons.airpersonsmobilehealth
20. Recipes by Ingredient. https://play.google.com/store/apps/details?id=com.abMobile.recipebyingredient
21. CVS/pharmacy. https://play.google.com/store/apps/details?id=com.cvs.launchers.cvs
22. Noom Weight Loss Coach. https://play.google.com/store/apps/details?id=com.wsl.noom
23. Drozer. https://www.mwrinfosecurity.com/products/drozer/
24. SnoreClock. https://play.google.com/store/apps/details?id=de.ralphsapps.snorecontrol
25. Sleep Talk Recorder. https://play.google.com/store/apps/details?id=com.madinsweden.sleeptalk
26. Urgent Care. https://play.google.com/store/apps/details?id=com.greatcall.urgentcare
27. Dex2jar. https://code.google.com/p/dex2jar/
28. WebMD mobile. http://www.webmd.com/mobile
29. Dixon P. Medical Identity Theft: The Information Crime that Can Kill You. World Privacy Forum; 2006 May 3.
30. Happtique suspends mobile health app certification program. http://mobihealthnews.com/28165/happtique-suspends-mobile-health-app-certification-program/
31. Symantec Corporation. How safe is your quantified-self? 2014 July.
32. Rooting – is it for me? Some Q&A. http://www.androidcentral.com/rooting-it-me-some-qa
33. Android historical version distribution. https://developer.android.com/about/dashboards/index.html