Information Leakage in Mobile Health Sensors and Applications

Anthony Louie

University of Illinois at Urbana-Champaign

Advised by Klara Nahrstedt

Senior Thesis

Abstract

Mobile health sensors and applications are at risk to information leakage due to the vulnerabilities present on mobile platforms and the risks of using wireless sensors. A possible vulnerability that has not been adequately researched in this area however is data leakage related specifically to how the sensor and the mobile device are designed interact with each other. Such vulnerabilities may exist because of how the health sensors are implemented through the operating system and how hardware is used in the devices. Through an analysis of a mobile health sensor we provide an idea of the current state of mobile health sensor security.

1. Introduction

We begin with an overview of mobile health sensors, including an explanation of why they have become more popular, and why security is becoming an important issue. Following that is a literature review that discusses key points of research related to this area. Next is the problem statement, which defines what this paper seeks to resolve. We then state the contributions that this paper makes in an effort to solve this problem. Finally there is an outline of the different sections contained in this paper.

1.1 Overview

Mobile health sensors are emerging as a technology and could potentially have a large impact on the wellbeing of humankind. Because of the universal need for maintaining one's health and the proliferation of the smart phone, the development of mobile health sensors provides a novel way for people to record and analyze their health information utilizing what they already have.

While mobile health sensing brings with it a lot of possibilities, it also presents several security concerns. These concerns stem from the nature of the data that is being collected and the platform the data is collected on. These sensors gather information related to a person's health, which is regarded as very sensitive data. In addition, these sensors operate and run applications on mobile devices, which have security concerns of their own. These concerns are increased by the fact that a mobile device is closely tied to the identity of an individual, as one does expect to share a mobile device like a computer. Because of these reasons, examining the security behind mobile health sensing is important.

1.2 Literature Review

There are many papers that discuss mobile security including mobile health security. Zhou et al. (2013) discuss how sensitive information on the Android platform can be leaked by accessing the operating system's public resources. One section of interest in this paper talks about how, by analyzing network traffic during the use of a medical mobile app, an attacker can identify what kinds of diseases were being accessed through the app. The paper challenges the design assumptions made on the Android platform so that more consideration is placed into reducing the risk of information leakage on the platform.

Yang et al. (2013) discuss their program analysis platform for detecting user-unintended data transmission on the Android platform. They make a distinction between user-intended data transmission and user-unintended data transmission, essentially a breach on privacy. The platform that they created is able to discriminate malicious apps seeking to steal information from other legitimate applications.

1.3 Problem Statement

While a considerable amount of research has been done on the subject of mobile security, including mobile health security, they primarily focus on either the security issues of the platform, or the security issues of the applications that run on them. This means that information can be leaked because of implementation flaws inherent in the operating system or because an application does not adequately protect its information.

We argue, however, that there needs to be research done on security issues related to using wireless medical sensors on a mobile platform, more specifically the possibility for health information leakage in how the sensor interfaces with the mobile device and its applications. In order to discuss this issue, we attempt to assess the current state of mobile health sensor technology by analyzing and classifying the vulnerabilities that exist in common mobile health sensors.

1.4 Contributions

The contributions that this paper makes are the following:

- We provide an understanding of major security issues in Android mobile health sensors.
- We test and verify possible information leakage from the use of a mobile health sensor.
- We create possible scenarios in which attackers take advantage of the vulnerabilities.
- We suggest a method for identifying potential vulnerabilities in a variety of sensors

1.5 Outline

We first provide context by giving an overview of the different mobile health applications and sensors currently being used. Following that will be an exploration of several different ways information can be leaked from using the sensors. We will show the vulnerabilities we found when using a particular sensor. During this discussion, we will describe scenarios that illustrate how these vulnerabilities could be exploited. We also describe a system of metrics we developed for identifying potential vulnerabilities in sensors. The following section discusses possible related topics for further research. In the last section we present the conclusions that we have made regarding mobile health sensors.

2. Background

In this section, where we give context for the issue of mobile health sensing security, we first introduce mobile health applications and give their categorizations and attack surfaces. After that is an introduction of the health sensors themselves, which includes a list of the different ways they are used, and how they can be categorized.

2.1 Mobile Health Applications

There is a lot of variety in the types of health applications available on the Android platform. He (2014) classifies these applications based on its target users and its functionalities. He categorizes around 9.38% of the most popular free mobile health applications on Google Play as medical sensor-based monitoring. The number of applications that make use of sensors, however, may be larger. Lifestyle management applications, which make up 60% of the applications classified, may also have some functionality that allows communication with mobile health sensors. An example of this would be the Fitbit application, which allows users to count calories and track eating and exercise habits as a lifestyle management application, but also allows connections with Fitbit sensors to monitor health statistics.

In addition to categorizing mobile health applications, He identifies several common attack surfaces for them. These include attacks on internet, third party services, Bluetooth, logging, SD card storage, and exported components. These attacks are very relevant to the security of using mobile health sensors, but its focus is not on the sensors themselves, but rather on the attacks that occur on applications. The exception to this point, however, would be attacks on Bluetooth, which would apply to any mobile health sensor that connects via Bluetooth. Bluetooth is one of the most common methods of communication for these sensors.

2.2 Mobile Health Sensors

Mobile health sensors measure a variety of statistics and are deployed in several different ways. There are commercial sensors that individuals can purchase for themselves, sensors that are used exclusively by health care providers as a part of their patient care programs, and additionally there are many new sensors currently being developed and researched. These sensors are not only different in how they are used and what kind of information they gather, but they also have different levels of risk. This is because of the different amounts of sensitivity in the information being collected and the amount of effort that is placed into securing the information.

Commercial sensors generally serve one out of two purposes. The first kind of sensor allows the user to monitor their fitness. These sensors include activity trackers, heart rate monitors, and weight scales. Activity trackers, one of the more popular health sensors that can be purchased, allow users to track their amount of “activity” by measuring the number of steps taken, distance traveled, calories burned. It measures motion related activities through the use of accelerometers that are worn on the body. Heart rate monitors are similar to activity trackers but instead of capturing motion, it counts the number of heart beats made within a certain timeframe. These sensors capture electrical signals of the body and are typically worn on the chest. Weight scales measure the user’s weight and possibly the user’s body composition. These allow users to automatically send data to their mobile device to keep track of over time.

The second purpose that commercial sensors can serve is to help users with their ongoing medical conditions. Blood pressure sensors help users with hypertension and blood glucose sensors can help patients with diabetes. These sensors are not only different in that they help with diseases rather than fitness, but that they often can fulfill their functions without the use of a mobile device,

but have the additional capability of storing data onto them. In addition, most of these sensors perform a single test, while many of the fitness sensors are designed to record data over a period of time.

Table 2.2 Types of commercial mobile health sensors

Category	**Type of sensor**
Fitness	Heart rate monitor
	Weight scales
	Activity trackers
Chronic illness management	Blood pressure
	Blood glucose
	Various others

3. Security Issues in Mobile Health Sensing

3.1 Methodology

In order to understand the security issues that exist in mobile health sensing, we examine three of its aspects. We first identify methods of attack on mobile health sensors. These methods are then classified into several different categories. Second, we then see how prevalent these vulnerabilities are with respect to different sensors. During this, we discuss that metrics we created in doing this. Third, we gauge how significant these threats are based on the information at risk and how widespread the issue is.

3.2 Potential Attack Surfaces

The first step in understanding security issues in mobile health sensing is to understand what the attack surfaces are. By knowing where the vulnerabilities exist, we can then classify them into categories. Splitting up the attacks into different types is useful not only for understanding the attacks themselves, but can provide a framework that can help with knowing where the source of additional vulnerabilities that we are not aware of.

3.2.1 Wireless Communication

The mobile platform is useful for health sensing is not only because of their widespread availability, but because mobile devices consist of components that make them useful for communicating and processing data. Communication protocols such as Bluetooth, Wi-Fi, and near field communication (NFC) can allow connections to external devices such as health sensors without the use of wires. This allows the end user to use sensors more easily, especially sensors that are worn on the body. However, this also represents a security concern as wireless communication can potentially be sniffed or injected into.

The most common way for health sensors to communicate with mobile devices is through Bluetooth. Security concerns with Bluetooth include denial-of-service, eavesdropping, man-in-the-middle, and modification. NIST has released a guide on Bluetooth security and it contains a list of

vulnerabilities for each version of Bluetooth among its information (Padgette 2012). Although each version of Bluetooth has its own security concerns, many improvements to Bluetooth security were made between versions 2.0 and 2.1. Using this list, one can see how many vulnerabilities may exist for a particular health sensor specifically because of its wireless communication.

3.2.2 Sensor Design

Health sensor vulnerabilities may also exist within the design of the sensor itself. Many sensors are not created with security in mind, and as a result, their design allows the leakage of information. We classify vulnerabilities that are a result of inherent design flaws as vulnerabilities resulting from sensor design. Whether this is because of an oversight or if it was the result of an intentional choice, as these sensors can be very singular in function, the sensor's design can become a liability. These vulnerabilities can be the result of side-channel attacks, the format or content of the messages being sent between the devices, or the use of insecure PINs or passwords.

Side-channel attacks on this platform can be used based on the size and frequency of messages being transmitted to the device. If the sensor communicates more frequently during some points of operation, or if the sensor communicates a larger amount of data for a particular reason, it may be possible for an attacker to infer what may be happening on the sensor by simply examining the size and frequency of traffic coming from the sensor. In addition, if the sensor uses a battery or some other power source, it may communicate its power usage, which can similarly be used as a side channel.

The other security issue with health sensors is that it may not take advantage of security mechanisms properly to safeguard its data. Because of the low processing power of many of the sensors, encryption may not be used, and as a result, data would be transmitted in plaintext. Some sensors rely on the assumption that sensors are only connected to devices that the user trusts. Other sensors may have very insecure security policies. This is also a problem because it is common for older Bluetooth devices to use very common and simple PINs that are hardcoded into the sensor.

3.2.3 Sensor Implementation

Health sensor vulnerabilities also may exist in how the sensor is implemented in software. This may be the result of the sensor API, the operating system's implementation of external devices, or how information about the sensor and data from the sensor is made available for different applications on the device.

Many sensors have a publically available API that allow anyone to develop an application that can work with their sensors. This gives users a larger selection of applications to use with their sensors. However, this represents a potential vulnerability. The Android platform provides its own public API for the Bluetooth Health Profile, a protocol designed by the developers of Bluetooth specifically for the use of health sensors. This protocol defines a minimum level of security in its security requirements (Wang 2012). However, the developers of the sensors may not have made their sensors compatible with these requirements or simply do not use this API. Because these

APIs do not make the same security guarantees as the Bluetooth Health Profile, they could potentially contain vulnerabilities.

Another potential attack surface in this area is that sensitive information can be leaked because sensors are accessed in a conventional way in which security is not considered. An example of this would be that in Android, you can access a list of names of all the connected Bluetooth devices from any application that has Bluetooth permissions. This mechanism allows applications to identify Bluetooth devices that they are compatible with, but this could also possibly be a violation of privacy.

For example, the attacker could create a database of the names of health sensors and compare it to the list of Bluetooth devices that are connected. If an attacker could identify the health sensors connected to the device, this could reveal what health conditions the user may be dealing with. A blood glucose sensor connected would indicate that the user is dealing with diabetes, for example. In addition, the same database used for identifying the connected sensor could be used to guess what applications are installed based on their compatibility with the sensor. This represents a vulnerability in that by knowing what software is installed, an attacker gains leverage for attacking the device.

Other concerns revolve around how the data of the sensor can be accessed. There may be issues if multiple applications can access the data, or if a single application is able to gain complete control of the sensor, denying service or modifying data. By answering the questions of how much control and information is given to the operating system and the applications running on the device, more vulnerabilities may be identified.

3.2.4 Application Attack Surfaces

Because mobile health sensors all interact with mobile apps, many of the possible attack surfaces related to them come from the applications that they interact with. As mentioned before, He categorized these attack surfaces as internet, third party services, Bluetooth, logging, SD card storage, and exported components. Although this paper focuses on the attack surfaces specific to the mobile health applications rather than mobile health sensors, this is too significant of a threat to not mention when discussing the security of mobile health sensors. Applications often present security concerns and they are a prime target of attack. Thus since they are often vulnerable and possess important information, their attacks surfaces are important threats in the space of mobile health sensing.

3.3 Pervasiveness of the Threats

After identifying the threats and types of threats that exist, the next step in order to understand the state of security in mobile health sensors is to identify how common the vulnerabilities that exist are within the health sensors that are currently being used.

We do this by first introducing the mobile health sensor that we initially investigated. We then share the information that we learned from its analysis. Then by combining this information with the categories we stated before, we develop metrics that can be used to help determine the

possibility of vulnerabilities in other mobile health sensors. We then apply these metrics to a small number of mobile health sensors we sampled in order to help complete our understanding of the pervasiveness of these vulnerabilities.

3.3.1 Zephyr HxM BT Heart Rate Monitor

The mobile health sensor that we initially investigated was the Zephyr HxM BT, a heart rate monitor. The sensor consists of a chest strap with sensors in contact with the skin, and a device that attaches to the strap that processes and communicates data. This fitness sensor connects to mobile devices running the Android and Windows phone operating systems through Bluetooth. The system is listed as being compatible with 23 Android applications and 6 Windows applications, and provides its own API for developers to use to make their own applications.

The sensor's measurements are heart rate, calculated using the R-R interval, speed, and distance. The heart rate is calculated with the following equation:

$$Instantaneous\ HR = 60\ /\ (RR\ interval\ in\ seconds)$$

In order to electronically measure the heart rate of a patient, typically the RR interval is used. The RR interval is the time between two R peaks in the waveform. These R peaks are part of the QRS complex, the periodic deflections visible on an electrocardiograph. The R peaks are the large upward deflection in as shown in the figure below.

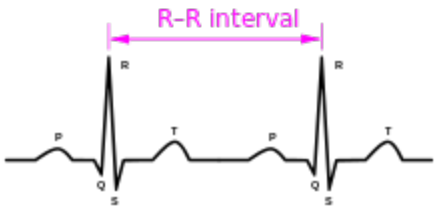

Figure of RR Interval. Source: Atkielski

The payloads from this sensor contain the times at which the previous 15 peaks were detected. The timestamps of these R peaks are represented by a 16 bit unsigned integer.

The speed measured by the sensor is the instantaneous speed of the wearer. The speed is measured in increments of 1/256 m/s, with a range of 0 to 15.996 m/s. The distance is the total distance traveled since the sensor was first turned on. It measures every 1/16 meters, with a range of 0 to 256m. Both of these values are represented by a 16 bit unsigned integer in the payload.

The complete structure of the data packet sent by this sensor can be found in Section 6, Table 6.9. When viewed in the Android operating system, the packet is represented by a byte array. This byte buffer is read into bytes, integers, and long integers depending on the data located at each offset as described in the table.

When deciding on the initial sensor, there were many factors involved in making the decision. An important factor for us was how many applications the sensor was compatible with, as a sensor with more applications would be more compelling to investigate. Because many of the health sensors and applications on mobile platforms are focused towards fitness, we decided on testing a fitness sensor. In addition, many of the chronic illness sensors are invasive and unsuitable for our own testing, or are standalone sensors that simply have extra features that allow communication with mobile devices.

When deciding between activity trackers, heart rate monitors, and scales, we chose heart rate monitors. We came to this decision because they were more mobile than scales and more singular in function, than activity trackers, meaning it would be easier to include in an application. While activity trackers and their applications could potentially be more interesting and deserve their own investigations, we felt that a heart rate monitor would be more transparent and suitable for our initial investigations. It is worthy to note that because activity trackers are very popular, many new products of this type are released often, and their applications advertise a variety of many features, it would make a very good subject for future analysis.

3.3.2 Sensor Analysis and Metric Forming

When analyzing the vulnerabilities of the Zephyr HxM BT, we made use of the categories for attack surfaces that we defined earlier to help guide our investigation. We will also present our findings using a similar outline.

Wireless communication: The Zephyr HxM BT makes use of Bluetooth to communicate to mobile devices. Given this information, we know that there may be security concerns regarding denial-of-service, eavesdropping, man-in-the-middle, and modification. Additionally, because we know that the sensor uses Bluetooth version 2.0, we can use the guide on Bluetooth security released by NIST and its list of vulnerabilities for each version to see what other known vulnerabilities may apply to the sensor. An important flag for security concerns is that the Bluetooth version is earlier than 2.1, the version in which many security features were implemented for Bluetooth communication. Thus we can designate this communication as being significantly unsecure because of its version. If a sensor has a higher version, then testing should be done to see if the new security mechanisms that were added are properly being used.

Sensor design: Examples of vulnerabilities categorized as results of sensor design are side-channel attacks, the format of the messages being sent between the devices, the use of insecure PINs or passwords, or other flaws in how the sensor was designed.

The specification for the Zephyr HxM raises several security concerns in how it was designed. The fact that it uses Bluetooth 2.0 could be considered a design flaw in itself, but because it additionally uses a common PIN ‘1234’ in a legacy pairing mechanism, it is quite easy to commit a denial of service or an interception of data. In addition, given an established Bluetooth connection to the sensor, there is no decryption needed to access the data. Given all of these security flaws, one could assume that for this particular sensor, security was not a concern in its design.

When investigating side-channel attacks, one considers whether the size or frequency of messages could potentially reveal any information. In the Zephyr HxM BT, messages have messages the fixed length of 60 bytes transmitted at a regular 1Hz interval. This means that because of the consistent stream of data, no information should be able to be learned from analyzing the size and frequency of messages. Additionally, there is information on the sensor battery being transmitted, but because they are transmitted alongside the health information being transmitted, you would not need to develop a side-channel attack in such a situation.

Sensor implementation: The Zephyr HxM provides its own API for developers to use to develop their applications. This means that the sensor does not implement the standard Bluetooth Health Profile, and none of its security mechanisms are guaranteed to exist for the proprietary API. This does not mean that necessarily there are vulnerabilities within this API, but it still represents the existence of an attack surface.

Another vulnerability we noticed while analyzing the Zephyr HxM was how easily the sensor could be identified, for both legitimate and possibly illegitimate reasons. It is simple to access this information in any application with Bluetooth permissions. You can retrieve a set of 'BluetoothDevices' using the 'getBondedDevices' method in the 'BluetoothAdapter' class. Then by using the 'getName' method on these devices, it may return a string that can identify any health sensors connected to the device.

```
BluetoothAdapter myBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();

Set<BluetoothDevice> myBondedDevices = myBluetoothAdapter.getBondedDevices();

if(myBondedDevices.size() > 0)

       for (BluetoothDevice device : myBondedDevices)

              if ( device.getName().startsWith("HXM") )

                     // Sensor is identified as a Zephyr heart rate monitor
```

In the case of the Zephyr HxM BT, if the string contained an 'HXM' in it, it would be considered to be a Zephyr heart rate monitor. Similar string identifiers would need to be developed for other sensors in order for this to work. In this case this information is not particularly harmful or sensitive. However, the same information leak could apply to a sensor in which its identification on a device could be significant. As in the example given before, the existence of a blood glucose sensor could point out the user's diabetes, which is a leak of information.

One other security issue we discovered was that only a single application could access the sensor at a time. Using this knowledge, an attacker could create an application that is loaded before the legitimate application, and effectively create a denial-of-service attack. There would be no way for the other application to get information from the sensor while the malicious application was controlling it. The malicious application must be forced to close by the user in order to free the sensor for other applications to use.

3.3.3 Mobile Health Sensor Survey

After examining one mobile health sensor, we can then use our findings to guide our assessment of other sensors for common vulnerabilities. We first list out the sensors that we used in our sample, and which categories they represent. We then describe what we learned from assessing the other sensors, and create a list of metrics, or indicators of possible vulnerabilities.

In our mobile health sensor survey, we only investigated commercially available sensors. We tried to have representatives from each category of health sensors listed previously. Health sensors were chosen based on how often different mobile health applications cite compatibility with the devices.

Table 3.3.3.1 Mobile Health Sensors Surveyed

Category	**Type**	**Sensor**
Fitness	Activity tracker	Fitbit Flex
		Jawbone Up24
	Heart rate monitor	Zephyr HxM BT
		Polar H7
	Weights and Scales	Withings Smart Body Analyzer
Chronic Illness management	Blood pressure	Homedics BPW-360BT
		A&D UA-767BT
	Blood glucose	MyGlucoHealth

The Fitbit Flex and Jawbone Up24 are both activity trackers worn on the wrist. These are the most popular health sensors on the market, used by people who want to live a healthier, active lifestyle. They use accelerometers to measure how much a person moves throughout a day. The main difference between these two sensors is the applications that they are coupled with. Another notable difference is that the Jawbone Up24 also can connect through a non-wireless interface.

The Zephyr HxM BT and the Polar H7 are both heart rate monitors that are worn on the chest. While similar in function, they are quite different in specification. The Zephyr used the older Bluetooth v2.0 while the Polar H7 used Bluetooth Smart, or version v4.0. Zephyr uses its own API while the Polar uses the Bluetooth profile provided by Bluetooth Smart for heart rate monitors.

The Withings Smart Body Analyzer, Homedics BPW-360BT, A&D UA-767BT, and MyGlucoHealth sensors are different than the previous sensors in that they typically make a single measurement whereas the other sensors are designed to be worn over a period of time. Many of these sensors are not reliant on a mobile device or wireless communication to operate, but rather have it as an added feature. As such, the sensors' support of mobile applications and mobile devices greatly varies. Some unique things about each sensor are that the Withings sensor also connects to devices through Wi-Fi, the Homedics sensor can connect to Microsoft HealthVault, a service designed to securely store health information, and the MyGlucoHealth sensor has been previously researched for vulnerabilities in its wireless communication by Mortazavi (2010).

For more detailed information on the mobile health sensors listed, please see the tables in Section 6 which contain more detailed information on each sensor.

While surveying different mobile health sensors, one of the first things checked was the version of Bluetooth the device used. Older Bluetooth versions were more vulnerable; they often had fixed PIN numbers and little guarantee of encryption during transmission. Most of the other sensors were Bluetooth version 4.0 or Bluetooth Smart. While these sensors did not have fixed PINs, they often were paired by pressing a button on the sensor, or by being near the receiving device, which is better than no security at all. The newer version of Bluetooth also has guarantees for encryption during communication, and provides health profiles for implementing different kinds of health devices. By using these health profiles an application can be assured of some level of security, and could mask the identity of the device being used.

One thing we learned during this survey is that some devices have security in place for who can access a sensor's APIs and who can make their applications compatible with the sensors. This is another security policy that is present, however not all sensors implement this policy.

The following table shows a few potential vulnerabilities one could look for. It is a summary of various vulnerabilities that one should consider when examining health sensors. Some metrics are very common but on the other hand do not reveal highly sensitive data. However, some metrics represent very fundamental security flaws and are still very common.

Table 3.3.3.2 List of Vulnerability Metrics and Their Threat Levels

Attack surface	**Metric**	**Threat Level**
Wireless communication	Bluetooth Version <2.1	High
	Unencrypted wireless communication	High
Sensor design	Common PINs or passwords	Medium
	Fixed PIN or passwords	Medium
	Proximity based pairing	Low
	Side-channel attacks	Medium
	Unencrypted storage	Medium
Sensor implementation	Nonstandard API	Low
	Unimplemented health profiles	Medium
	Denial of Service	Medium
	Ability to identify sensor	Low
	No developer authentication	Medium

3.4 Significance of the Threat

The data that can be generated by health sensors can greatly vary in sensitivity. Some health information can be very embarrassing or can have significant effects on a person's life if revealed. However, a lot of information measured by mobile health sensors is simple, or is not particularly sensitive, at least in the user's perspective.

There is also the question of whether or not the security of sensors is proportional with the sensitivity of the information it generates. A heart rate monitor might not value security as much as a sensor used by a health care program to monitor of its patients. One might be able to argue that manufacturers of these sensors implement only as much security as its functions demand, and that this is not a significant issue. However, if this is not followed in real situations, then it represents an important problem.

Mobile platforms have had several incidents in the past where applications intended to steal personal information about users to sell for profit. In an attempt to reduce the amount of information leakage on mobile platforms, mobile operating systems give users more information about the resources that applications are allowed to access. However, this only provides security if the user pays close attention to the applications that are installed.

4. Topics for Further Research

The investigations that were carried out in this study were still quite preliminary, and as a result, there are many topics for further research.

We came up with four different categories of attack surfaces, but there could possibly be more. The categories that we defined can also be further split into more specific definitions. By doing this we would add to the framework we can use to do research in mobile health sensing security.

We tested and verified vulnerabilities in a single sensor, but there can be a lot more to be learned from testing other sensors. It would give us insight into the different categories of sensors, and help us compare and contrast their differences. Investigations of other sensors would also generate more metrics, as many of ours were formed during the analysis of our sensor. More investigations also gives us a larger sample size, helping us understand more about this field. And of course, any novel attacks or vulnerabilities that affect mobile health sensors in general would be a very significant discovery in this field.

A very interesting category to research would be activity trackers, as they are part of a fast-moving field in consumer technology and can be more complex and unique than some of the single-function sensors. Noncommercial sensors could also be compelling research material, as they may provide more advanced features, generate more sensitive data, and have security policies that they must fulfill.

5. Conclusions

Mobile health sensing has so much potential for impacting people's lives in the future. We may see them become deeply integrated in how we manage our bodies. However, we have seen and explored the possible ways that security may be an issue. Not only in how secure mobile operating systems are, or how safe the mobile health applications are, but also in the health sensors themselves.

The security of fitness sensors, like activity trackers, heart rate monitors, and scales, as well as sensors used for illnesses, such as blood glucose and blood pressure sensors, could analyzed for potential attack surfaces in how they communicated, how they were designed, and how they were implemented. These attack surfaces were then used to guide our investigations, during which we developed metrics for indicating possible vulnerabilities. Then finally after factoring in how significant the current threats were, we could develop a better understanding of mobile health sensor security.

6. Tables and Data

Table 6.1.1 Fitbit Flex Details

Category	**Details**
Measurements	Steps, Distance, Calories burned, Duration of activity, sleep duration, movement during sleep
Bluetooth Version	Bluetooth Smart (v4.0)
Wireless Encryption	Encrypted
Pairing mechanism / PIN	Proximity and sensor activation
Application Authentication	OAuth 1.0
API	Fitbit API (Proprietary)
Message Frequency	On API request
Message size	Dependent on request type and user data
Message type	GET, POST, JSON, XML
Other	Can upload data to Fitbit servers and can also share data with third parties.

Table 6.1.2 Fitbit Flex Attack Surfaces and Metrics

Attack surface	**Metric**	**Vulnerable?**
Wireless communication	Bluetooth Version <2.1	No
	Unencrypted wireless communication	No
Sensor design	Common PINs or passwords	N/A
	Fixed PIN or passwords	N/A
	Proximity based pairing	Yes
	Side-channel attacks	Yes
Sensor implementation	No developer authentication	No
	Nonstandard API	Yes
	Unimplemented health profiles	Yes

Table 6.2.1 Jawbone Up24 Details

Category	**Details**
Measurements	Distance, Calories burned, Duration of activity, activity intensity, sleep duration, movement during sleep
Bluetooth Version	Bluetooth Smart (v4.0)
Wireless Encryption	Encrypted
Pairing mechanism / PIN	Proximity and sensor activation
Application Authentication	OAuth 2.0
API	UP for Developers (Proprietary)
Message Frequency	On API request
Message size	Dependent on request type and user data
Message type	GET, POST, JSON
Other	Can also sync through Simple Sync, its non-wireless interface

Table 6.2.2 Jawbone Up24 Attack Surfaces and Metrics

Attack surface	**Metric**	**Vulnerable?**
Wireless communication	Bluetooth Version <2.1	No
	Unencrypted wireless communication	No
Sensor design	Common PINs or passwords	N/A
	Fixed PIN or passwords	N/A
	Proximity based pairing	Yes
	Side-channel attacks	Yes
Sensor implementation	No developer authentication	No
	Nonstandard API	Yes
	Unimplemented health profiles	Yes

Table 6.3.1 Zephyr HxM BT Details

Category	**Details**
Measurements	Heart rate, Distance, Speed
Bluetooth Version	Bluetooth v2.0
Wireless Encryption	Unencrypted
Pairing mechanism / PIN	Bluetooth Legacy pairing. Common PIN '1234'
Application Authentication	None
API	Zephyr API (Proprietary)
Message Frequency	Constant
Message size	Constant
Message type	Serial Data
Other	None

Table 6.3.2 Zephyr HxM BT Attack Surfaces and Metrics

Attack surface	**Metric**	**Vulnerable?**
Wireless communication	Bluetooth Version <2.1	Yes
	Unencrypted wireless communication	Yes
Sensor design	Common PINs or passwords	Yes
	Fixed PIN or passwords	Yes
	Proximity based pairing	No
	Side-channel attacks	No
Sensor implementation	No developer authentication	Yes
	Nonstandard API	Yes
	Unimplemented health profiles	Yes

Table 6.4.1 Polar H7 Details

Category	**Details**
Measurements	Heart rate
Bluetooth Version	Bluetooth Smart (v4.0)
Wireless Encryption	Encrypted
Pairing mechanism / PIN	Proximity and sensor activation
Application Authentication	None
API	Bluetooth Heart Rate Profile
Message Frequency	Constant
Message size	Constant
Message type	Serial Data
Other	None

Table 6.4.2 Polar H7 Attack Surfaces and Metrics

Attack surface	**Metric**	**Vulnerable?**
Wireless communication	Bluetooth Version <2.1	No
	Unencrypted wireless communication	No
Sensor design	Common PINs or passwords	N/A
	Fixed PIN or passwords	N/A
	Proximity based pairing	Yes
	Side-channel attacks	No
Sensor implementation	No developer authentication	Yes
	Nonstandard API	No
	Unimplemented health profiles	No

Table 6.5.1 Withings Smart Body Analyzer Details

Category	**Details**
Measurements	Weight, Body fat percentage, heart rate, carbon dioxide levels in air
Bluetooth Version	Bluetooth Smart (v4.0)
Wireless Encryption	Encrypted
Pairing mechanism / PIN	Proximity and sensor activation
Application Authentication	OAuth 1.0
API	Withings API (Proprietary)
Message Frequency	On API request
Message size	Dependent on request type and user data
Message type	GET, POST, JSON
Other	Can also connect to Wi-Fi devices

Table 6.5.2 Withings Smart Body Analyzer Attack Surfaces and Metrics

Attack surface	**Metric**	**Vulnerable?**
Wireless communication	Bluetooth Version <2.1	No
	Unencrypted wireless communication	No
Sensor design	Common PINs or passwords	N/A
	Fixed PIN or passwords	N/A
	Proximity based pairing	Yes
	Side-channel attacks	Yes
Sensor implementation	No developer authentication	No
	Nonstandard API	Yes
	Unimplemented health profiles	Yes

Table 6.6.1 Homedics BPW-360BT Details

Category	**Details**
Measurements	Blood Pressure (systolic, diastolic), heart rate
Bluetooth Version	Bluetooth Smart (v4.0)
Wireless Encryption	Encrypted
Pairing mechanism / PIN	Proximity and sensor activation
Application Authentication	None
API	None
Message Frequency	On measurement
Message size	Varies on number of new readings
Message type	Not available
Other	Can send results to Microsoft Healthvault

Table 6.6.2 Homedics BPW-360BT Attack Surfaces and Metrics

Attack surface	**Metric**	**Vulnerable?**
Wireless communication	Bluetooth Version <2.1	No
	Unencrypted wireless communication	No
Sensor design	Common PINs or passwords	N/A
	Fixed PIN or passwords	N/A
	Proximity based pairing	Yes
	Side-channel attacks	Medium
Sensor implementation	No developer authentication	Yes
	Nonstandard API	Yes
	Unimplemented health profiles	Yes

Table 6.7.1 A&D UA-767BT Details

Category	**Details**
Measurements	Blood Pressure (systolic, diastolic), heart rate
Bluetooth Version	Bluetooth v2.1
Wireless Encryption	Yes
Pairing mechanism / PIN	Common PIN '123456'
Application Authentication	None
API	None
Message Frequency	On measurement
Message size	Constant
Message type	Serial data
Other	

Table 6.7.2 A&D UA-767BT Attack Surfaces and Metrics

Attack surface	**Metric**	**Vulnerable?**
Wireless communication	Bluetooth Version <2.1	No
	Unencrypted wireless communication	No
Sensor design	Common PINs or passwords	Yes
	Fixed PIN or passwords	Yes
	Proximity based pairing	No
	Side-channel attacks	Low
Sensor implementation	No developer authentication	Yes
	Nonstandard API	No
	Unimplemented health profiles	Yes

Table 6.8.1 MyGlucoHealth Details

Category	**Details**
Measurements	Blood glucose
Bluetooth Version	Bluetooth v2.0
Wireless Encryption	None
Pairing mechanism / PIN	Common PIN (0000)
Application Authentication	Developer Verification required to access API
API	MyGlucoHealth Wireless Developer's Kit
Message Frequency	On measurement
Message size	Constant
Message type	Serial
Other	Also connects USB to PC. Has online web portal. Previously hacked by Mortazavi (2010).

Table 6.8.2 MyGlucoHealth Attack Surfaces and Metrics

Attack surface	**Metric**	**Vulnerable?**
Wireless communication	Bluetooth Version <2.1	Yes
	Unencrypted wireless communication	Yes
Sensor design	Common PINs or passwords	Yes
	Fixed PIN or passwords	Yes
	Proximity based pairing	No
	Side-channel attacks	Low
Sensor implementation	No developer authentication	No
	Nonstandard API	Yes
	Unimplemented health profiles	Yes

Table 6.9 Zephyr HxM Data Message Structure

Byte	Data type	Data	Data Values/Ranges
0	Byte	STX Start of Text	0x02 Delimiter
1	Byte	Message ID	0x26 Identifies HxM packet
2	Byte	Data Length Code	0x37 Payload length of 55
3	Integer	Firmware ID	Firmware type
5	Integer	Firmware Version	Firmware version
7	Integer	Hardware ID	Hardware identity
9	Integer	Hardware Version	Hardware version
11	Integer	Battery Charge Indicator	0 to 100% charge
12	Integer	Heart Rate	30 to 240 bpm
13	Integer	Heart Beat Number	0 to 255 beats, rolls over
14	Long	Heart Beat Timestamp 1-15	0 to 65535 ms, rolls over
44	Long	Reserved	Reserved
50	Long	Distance	0 to 4095 1/16m, rolls over
52	Long	Speed	0 to 4095 1/256m/s, rolls over
54	Byte	Strides	0 to 128, rolls over
55	Byte + Long	Reserved	Reserved
58	Byte	CRC	CRC calculated from payload
59	Byte	ETX End of Text	0x03 Delimiter

PseudoCode for reading data:

```
byte[] msg; // HxM Message

int index = 0;

byte stx = msg[index++];

byte msgID = msg[index++];

... // Do for all data in message

byte etx = msg[index];
```

7. Acknowledgement

We acknowledge support from the National Science Foundation through the Information Trust Institute (ITI) and the Illinois Cyber Security Scholars Program during the Spring 2014 semester for this research.

This work has also utilized the NSF TWC 1329686 funding on 'Enabling Trustworthy Cyber-systems for Health and Wellness'.

The work also benefited from NSF CNS 13-30491 (THaW).

All work in the senior thesis is my own and does not reflect the views of the funding agencies.

8. References

1. Padgette, John, Karen Scarfone, and Lily Chen. Guide to Bluetooth Security: Recommendations of the National Institute of Standards and Technology. Revision 1. 2012. Web. March 2014. <http://csrc.nist.gov/publications/nistpubs/800-121-rev1/sp800-121_rev1.pdf>.

2. Zhou et all. Identity, Location, Disease and More: Inferring Your Secrets from Android Public Resources. New York, NY. ACM. 2013. Proceeding.

3. He, Dongjing. Security Threats to Android Apps. University of Illinois at Urbana-Champaign. 2014.

4. Yang et all. AppIntent: Analyzing Sensitive Data Transmission in Android for Privacy Leakage Detection. New York, NY. ACM. 2013. Proceeding.

5. HxM Bluetooth API Guide. Version 1.7. 2010. Web. March 2014. <http://zephyranywhere.com/media/pdf/HXM1_API_P-Bluetooth-HXM-API-Guide_20100722_V01.pdf>

6. Wang, et all. Health Device Profile (HDP). Revision V11. 2012. Web. March 2014. <https://www.bluetooth.org/docman/handlers/DownloadDoc.ashx?doc_id=260864&vId=290095>.

7. Atkielski, Anthony. ECG-RRinterval.svg. 2009. Wikimedia Commons. Web. March 2014. < http://commons.wikimedia.org/wiki/File:ECG-RRinterval.svg>

8. Mortazavi, Bobak, and Sarrafzadeh, Majid. Hacking a Medicial System: A Method for Intercepting and Corrupting Signals of a Bluetooth Glucometer. Oct 2010. Embedded and Reconfigurable Systems Laboratory UCLA. Web. March 2014. <http://www.cs.ucla.edu/~bobakm/Publications_files/Glucometer%20Security%20White%20Paper.pdf>