2018

Vulnerability analysis: protecting information in the IoT

Brian Cusack

Feiqiu Zhuang

VULNERABILITY ANALYSIS: PROTECTING INFORMATION IN THE IoT

Brian Cusack, Feiqiu Zhuang
Cyber Forensic Research Centre, Auckland University of Technology, New Zealand
brian.cusack@aut.ac.nz, feiqiu.zhuang@aut.ac.nz

Abstract

The research was designed to study IoT security vulnerabilities and how to better protect IoT communications. By researching the system a Fitbit uses for communications, this research analyzes and reveals security defects in the IoT architecture. The research first uses a man-in the middle (MITM) attack to intercept and analyze the Fitbit system traffic to identify security weakness. Then uses a replay attack to further validate these flaws. Finally, countermeasures against these security threats are proposed. The research findings show the Fitbit's IoT communication architecture has serious information security risks. Firstly, the Fitbit tested does not encrypt the raw data between the mobile app and Fitbit servers. It uses HTTPS to secure communication between the mobile phone and the Fitbit servers. Once HTTPS is broken, all raw data can be read and tampered with. Secondly, Fitbit uses Base64 credentials to associate the Fitbit tracker, and Fitbit app with the Fitbit user account. Base64 can be easily broken on the Internet or using other tools. Attackers can generate fake Base64 credentials to hack a user account. According to the experimental results from the study, the IoT should secure every node in its architecture. It is also necessary to encrypt the raw data and not just rely on HTTPS. It is recommended to replace the Base64 algorithm with AES and hashing.

Keywords
Security, Failure, IoT, Information, Disclosure

INTRODUCTION

We have reported security vulnerabilities in wearable devices and their Information Systems elsewhere but in this paper we focus on one specific device and its IoT context to identify the specific security vulnerabilities. Research on the vulnerabilities in the wearable Fitbit device have been reported by others (for example, Fereidooni, T. Frassetto, M. Miettinen, A., Sadeghi, M., Conti, H., 2017, "Fitness Trackers: Fit for health but unfit for security and privacy") but we replicate and extend this work by identifying the multilayered provisioning of services that lead to the reuse of information by the related parties. The targeted device and Information System is indicative of other IoT devices and the information architecture security risks. A fitness tracker uploads data to the mobile phone via Bluetooth and the mobile phone runs the Tracker App that communicates with the Tracker server via WIFI. Between the mobile phone and tracker remote sever, are vulnerabilities where man in the middle (MITM) proxies may be implemented to compromise and disclose personal information. The information architecture also permits reuse of information by related parties for revenue and intelligence gathering. Classen, Wegemer, Patras, Spink, and Hollick (2018) also report wearables' security to be vulnerable to MITM attack. They undertook reverse-engineering to prove how a replay attack compromises the security of Fitbit trackers. The user account of a Fitbit tracker can be re-associated to one attacker account using authentication replay. The reason is that a Fitbit App runs HTTPS to secure traffic between the mobile and the Fitbit server but HTTPS is too weak to block a MITM attack.

Cyr, Horn, Miao, and Specter (2014) used Charles Proxy (one MITM tool) to analyse transport layer security (TLS) traffic between the Fitbit App and the Fitbit Server. They intercepted the Bluetooth Low Energy (BTLE) key which is used to encrypt communication between the Fitbit Tracker and the Tracker App (mobile App). By using the Charles Proxy, they were able to decrypt and view in plain text the traffic between the mobile app and the Fitbit server, which contained all the user data. Fereidooni, Classen, et al. (2017) also applied the MITM proxy to hack Fitbit devices. They first used Wireshark to capture and analyse packets passing through the MITM proxy. Then they tampered the Fitbit's data to alter the user performance. They successfully submitted fake data to the Fitbit server, causing the mobile app to display incorrect statistics to the user (and those reusing the information). To make the experiments work, the researchers created a WIFI hotspot on a laptop which ran a MITM proxy (Figure 1 shows the system architecture for evaluating the Fitbit communication system as described in the literature). The mobile phone chose to connect with this hotspot. In this way, all the traffic went to the Internet from the mobile phone and it must go through the MITM proxy first. The MITM proxy needs to act as one certificate authority (CA), and the mobile phone needs to install that certificate issued by the MITM proxy. If no

certificate is installed on the mobile, it is not possible to monitor, intercept, and analyse the data between the mobile phone and the server because the Tracker Apps use SSL/TLS (https) to synchronize data with the remote server. In this paper we provide an overview of IoT security, the experimental set up, the results, and a brief discussion of the implications of these findings for countermeasures.

IOT SECURITY

The hardware of an IoT device is usually composed of the sensor, actuator, and built-in communication module (Gubbi, Buyya, Marusic, & Palaniswami, 2013). IoT devices use two connection modes to collect sensor data from monitored targets. One is a wireless connection, such as ZigBee, Bluetooth, or wi-fi, and another mode is wired connection, such as Ethernet (Tilkov, 2015). Hahm, Baccelli, Petersen, and Tsiftes (2016) classified IoT devices into 2 categories based on their operating system (OS): high-end IoT devices and low-end IoT devices. High-end IoT devices can run traditional OS such as Microsoft, or Linux. Low-end IoT devices can only run lightweight or customised OS. The performance of IoT devices normally depends on energy capacity, memory, and CPU. The physical size of the IoT device is generally small. Its memory and CPU performance lags far behind other common computers and laptops and it is not practical to run complex software and perform complex computations (Gaur & Tahiliani, 2015). Amadeo et al. (2016) also maintained IoT devices not only have severe restrictions on energy and computing, but also on the networking features. Trappe, Howard, and Moore (2015) pointed out the low-end IoT devices are relatively inexpensive, which is why they have restrictions on energy, computation, and storage. This low energy and low computing model limits IoT devices to run some high-level applications such as conventional encryption algorithms, and asymmetric encryption is not generally suitable for resource-constrained IoT devices. For example, RFID tags do not support RSA 1024 installation, and asymmetric encryption uses longer encryption keys which consumes the memory and power of low resource IoT devices very quickly.

The IoT architecture proposed by Wu, Lu, Ling, Sun, and Du (2010) is generally accepted. This architecture contains three layers: perception layer, network layer, and application layer. The perception layer includes IoT devices and monitored objects. It provides the raw data resource and it uploads data to the network layer. The network layer is responsible for the functionalities of transmit, store, and analyse the data, from the perception layer. The application layer outputs the processed raw data and makes it readable to the end users. Gubbi et al. (2013) say the IoT has three components: IoT device, middleware, and presentation. IoT devices generate sensor data. Middleware stores and analyses the sensor data, such as a third-party Cloud platform. The Presentation component provides for the different terminal applications, such as mobile Applications and a web dashboard. Based on these three elements, they divided IoT architecture into 3 layers. The top layer is the application layer, which runs different client applications, such as environment monitoring, and health monitoring. The middle layer is the Cloud computing which provides storage and data analysis. The bottom layer is the wireless sensor network, including IoT devices and network devices. These architectures and hardware arrangements provide the IoT context and the context for security evaluation.

Analysis shows, IoT attacks can happen in the perception layer, network layer, and application layer. Farooq, Waseem, Khairi, and Mazhar (2015) described the security risks at different IoT layers. Table 1 summarises these risks.

Table 1. Security Risks at IoT layers

IoT layer	Security Risk
Application layer	DDoS, malicious data injection, sniffing， and phishing
Network layer	Distributed Denial of Service (DDoS) attacks, ARP poisoning, MITM
Perception layer	unauthorized access, mac spoofing, radio hijacking， and eavesdropping

Others identify security risks at the different layers (for example, Mahmoud, Yousuf, Aloul, and Zualkernan, 2015). They suggest the Perception layer, is vulnerable to replay attack, timing attack and DoS attack. The replay attack is mainly conducted by spoofing, eavesdropping, or tampering with the data of IoT devices. The timing attack can destroy the encryption system by analysing the time it takes to execute the encryption. The DoS attack is to rapidly deplete IoT energy and increase its load to paralyse the whole IoT network. At the Network layer, they assert Dos and MITM attack is the main risk. On the application layer, they show the risk is from insecure authentication mechanisms and privacy breaches. Because users cannot manage what data they want to reveal, they also do not know how their

data will be used. Rahman, Daud, and Mohamad (2016) also raised eight security threats to IoT. The first threat is from the insecure application programming interface (API) interfaces, such as insecure encryption, plaintext authentication, reusable tokens, and unencrypted data. The second threat is from inadequate authorization, such as the use of one improper authentication or one low security authorization. The third threat is from insecure network connections, such as using of HTTP or no encryption tunnel between IoT nodes. The fourth threat is from unencrypted raw data, such as transmitting plaintext data over the network. The fifth threat is privacy vulnerability where the IoT host services reuse data and the users do not know what data will be collected, they cannot control what data they want to disclose, and they do not know of the vendor contracts and other reuse/sale of their data. The sixth threat is from inadequate encryption methods, for example, when one insecure or low security encryption algorithm is applied. The seventh threat is from software or firmware defects, for example, bugs in software and firmware, and IoT vendors do not patch these bugs in a timely manner. The eighth threat is from insecure physical accesses, for example, anyone can physically access IoT devices, that do not have strong physical access management. Hackers can also inject malicious code or software into IoT devices directly.

TEST SET UP

Many research papers discuss IoT security vulnerabilities. However, few research articles specifically explain and validate these security vulnerabilities based on IoT architectures through experiments. For example, the researchers cited above identify theoretical security vulnerabilities of IoT in various layers, but they did not collect empirical data to show what the specific security defects are, and what the consequences of these defects can be. One reason may be IoT does not yet have a standard architecture and it is difficult to define a set of standard IoT security vulnerabilities. Unlike the mature Open Systems Interconnection (OSI) 7- layer model, where each of its layers has a clear and open definition of security vulnerabilities. Another reason is IoT is a heterogeneous connection involving different technologies, such as sensor technology, Bluetooth, Wi-Fi , 4G or 5G, and Cloud computing. It is a big challenge to analyse IoT security vulnerabilities at all levels. To help IoT users and other researchers to more intuitively understand security vulnerabilities faced by IoT this experimental research examined IoT security vulnerabilities at IoT application layers through experimenting on one leading IoT device and its communication system. The security vulnerabilities of this IoT system are demonstrated by intercepting, analysing, and testing its traffic.

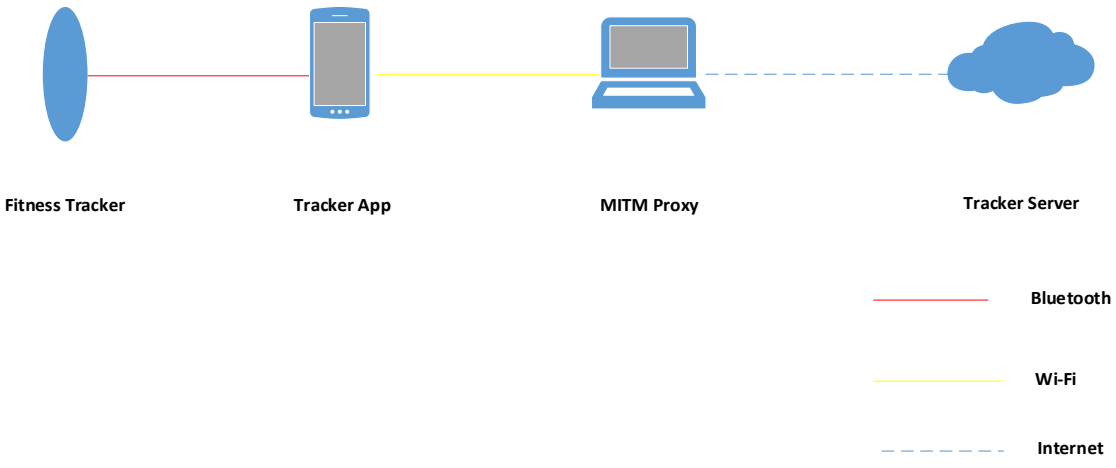

Figure 1. Theoretical Research Architecture from the Literature

The theoretical research reviewed all suggest the architecture shown in figure 1 is the standard information connection. However, in practice this architecture overlooks important feedback loops and interaction between the key elements. Consequently, we propose figure 2 as a relevant architecture for experimental testing for potential security vulnerabilities. Figure 2 also shows the actual IP of the equipment used. Figure 3 outlines the phases in the research. In phase 1 we had to take the literature suggestions and turn them into a working environment for testing. This included selecting the hardware, the network configurations, and the relevant software. In phase 2 the traffic between the Android phone and the Fitbit server was found to be vulnerable. It was captured, analysed and

manipulated. In phase 3 the experiments were repeated using an iPhone, and in phase 4 the experimental data was analysed in order to propose countermeasures that can help mitigate the risks.

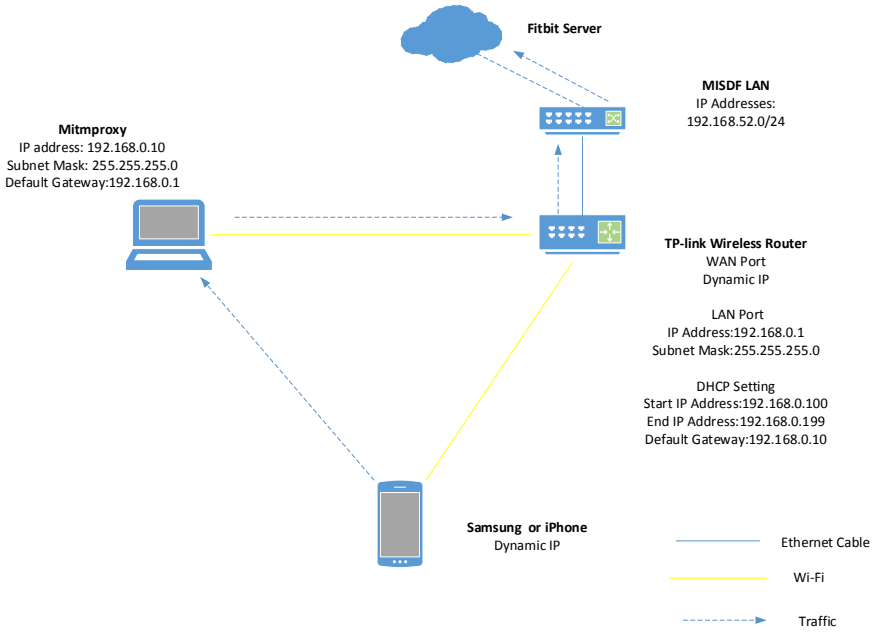

Figure 2. Modified Architecture for Experiments

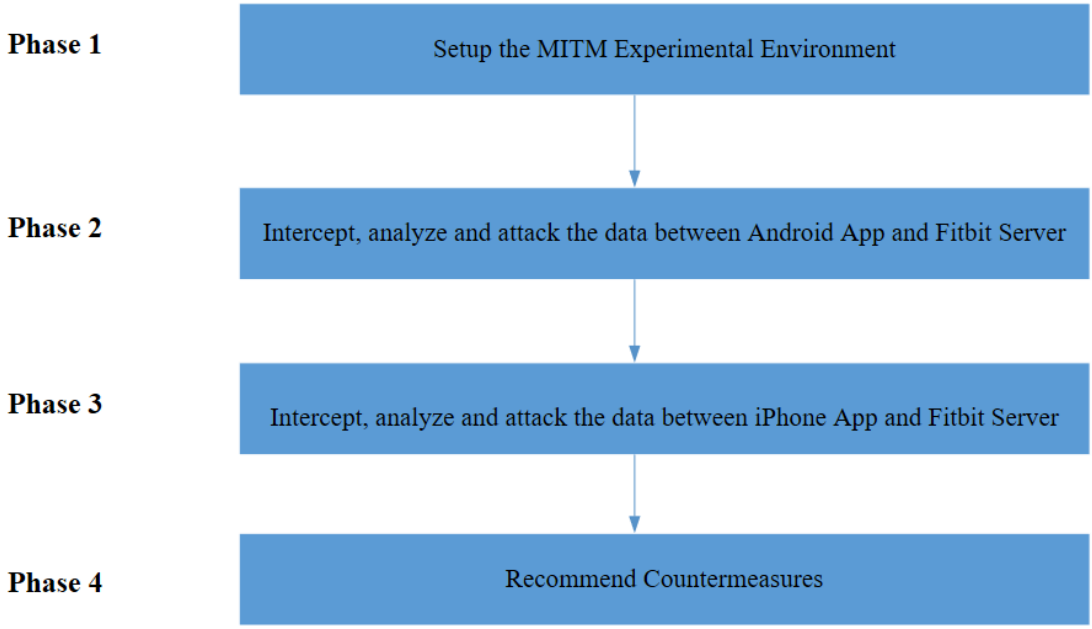

Figure 3. Research phases

RESULTS

Fitbit information system was evaluated and the vulnerabilities targeted for exploitation. Traffic was analysed and then a number of techniques used to try and break it by gaining plain text. The major vulnerability identified is between the phone and the server. Many security vulnerabilities were discovered by analysing the Fitbit traffic but the greatest weakness in the Information System is that Fitbit does not encrypt most of the data between the mobile application and the Fitbit servers. It is exchanged in plain text. The plaintext not only shows personal

fitness and health data, but also personal and phone information. In addition, Fitbit uses the Base64 algorithm to authenticate and authorize Fitbit users. Base64 encoding is designed to convert binary to characters. It does not provide encryption because it does not require any encryption keys and the algorithm is publicly available. The major finding in the testing was that the Base64 communications could be turned into plain text. The request data and response data are readable because they are in plaintext. Some data is difficult to read because it is encrypted with AES. However the Bluetooth connection is weak and this data can be compared with the data the Fitbit application uploads to the Fitbit server. Figure 4 shows the result of a compromise where the plain text between the phone and the server is intercepted, altered and then sent on to the server. On the left hand side the fitness user is counted by the device as doing 5761 steps, the link between the phone and the server is compromised and now it shows 50000 steps. Such erroneous data can be generated for heart beats and any other data attribute that the device generates, exposing the user to information vulnerability.

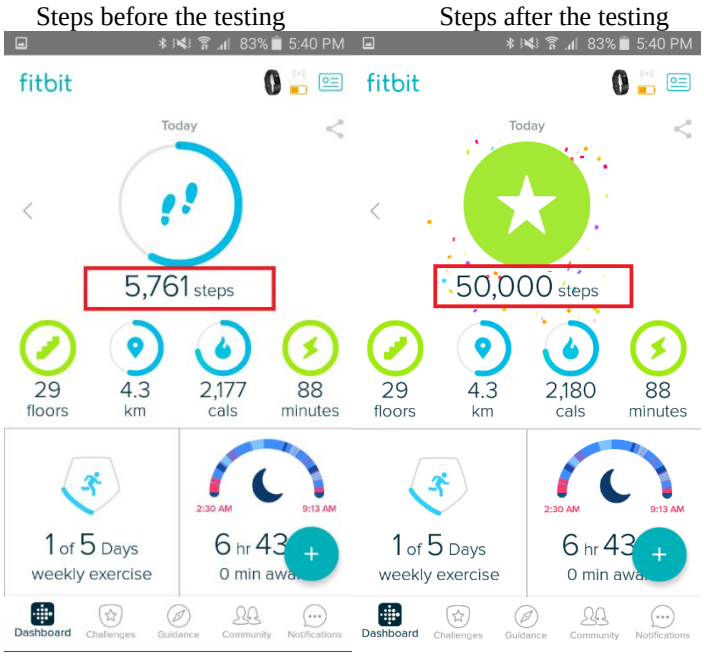

Figure 4. Compromised Fitness Data

Figure 5 identifies each of the Fitbit information exchanges and Table 2 lists each information exchange vulnerability.

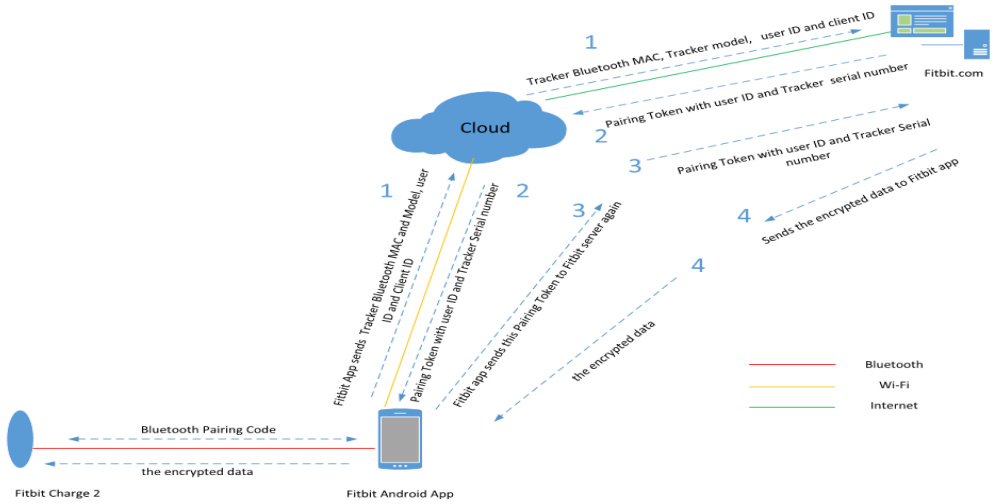

Figure 5. Each Information Exchange from Fitbit to Server

Table 2. Identified Security Vulnerabilities in Testing

Information Exchange	**Vulnerability**
Tracker Fitbit association	Bluetooth
Phone App to analytics.com	Privacy, Information reuse
Phone App Token	Refresh rate (8 hours), Base64 weakness
Phone App to mixpanel.com	Privacy, information reuse
Data dump to crashlytics.com	Privacy, information reuse
Tracker megadump	Privacy, information reuse
Phone status data	Can be read in plain text
Phone App to Facebook	Privacy, information reuse
Phone App requests Tracker status	Discloses MAC, Base64 weakness
Fitbit requests device type	Full disclosure, hackers dream
Fitbit requests user profile	Full disclosure, privacy issue
Fitbit requests tracker authentication	Man in the middle opportunity
Fitbit requests firmware synchronization	Appears to have AES encryption but a fixed 8 bit word. Potential to compromise.
Fitbit phone monitoring	Base64 weakness for all data

COUNTERMEASURES

Fitbit Information System has several places to improve its security measures. The first scheme is to encrypt the request data from the mobile application and the response data from the Fitbit server, and not to depend on a TLS

tunnel. For example, it sends mobile phone information to crashlytics.com, facebook.com and Fitbit server in plaintext, and thereby directly exposes the tracker's serial number, the user's profile data, fitness and health data from the Fitbit server. An attacker can sell the response data to make profits. A compromise in the request data can cause Fitbit users to lose connection with the Fitbit tracker. The leakage of the response data can result in the exposure of the user's private data and open other vectors for an attacker. Another measure is to implement secure encryption methods to replace the Base64 algorithm. A tampered user ID can make authorization credentials invalid. The authorization credential is valid for one Fitbit user within 8 hours if this user uploads the correct authorization credential to the Fitbit server. But due to this, the Fitbit application must repeatedly generate a new authorization credential until the Fitbit server accepts its credentials. The process is open to abuse and also discloses a point of entry for an attacker. Given that the Base64 algorithm is easy to compromise using local or online tools, it is not a good choice for large-scale deployment in business. Hashing can be used to replace Base64 to make it difficult to attack. Different from Base64, the hash value cannot be reversed and it is impossible to compute the original plaintext based on the hash value. The client ID and client password of a Fitbit user never change and can be hashed. The client ID and the user ID use the Base64 algorithm, which is why they were compromised in the experiment. If the client ID and the user ID are hashed, an attacker will face great difficulties compromising the Information System.

For a static parameter the Fitbit server just needs to compare whether these hash values match the ones stored in its database. If they are fully identical, the Fitbit server will grant access. If not, Fitbit will reject the request. The advantage of this countermeasure is a hacker cannot understand the meaning of these hash values because they are irreversible, unlike Base64. For other dynamic plaintext data such as a pairing token, "authSubkey" and "nonce", these data need to be changed frequently and uploaded. AES is a good choice for encrypting them. Based on the paper by Classen et al., (2018), each Tracker is installed with one AES128 symmetric key when it comes out of the factory. The tracker applies this AES128 key to encrypt microdump and megadump then uploads them to the Fitbit server. The proposed solution is that the Fitbit Application should also obtain this AES128 key after the tracker, user ID, and client ID are authenticated and bundled together. Fitbit app can use this AES128 key to encrypt the request data and decrypt the response data once it obtains the AES128 key. This solution means Fitbit needs to develop a mobile application which supports AES encryption and decryption. It may take further design but it is worth investing if considering the product image, user confidence and the possible litigation from the users. A more secure solution is to apply another different AES128 key between the Fitbit Application and Fitbit server. This solution model needs Fitbit servers to firstly decrypt the microdump and megadump from the Fitbit mobile application, and then encrypt the microdump and megadump again using a new AES128 key. This solution avoids all data being exposed if the single AES128 key is compromised. In this proposed solution, the Fitbit server needs to decrypt and encrypt the same data twice, which is also efficient.

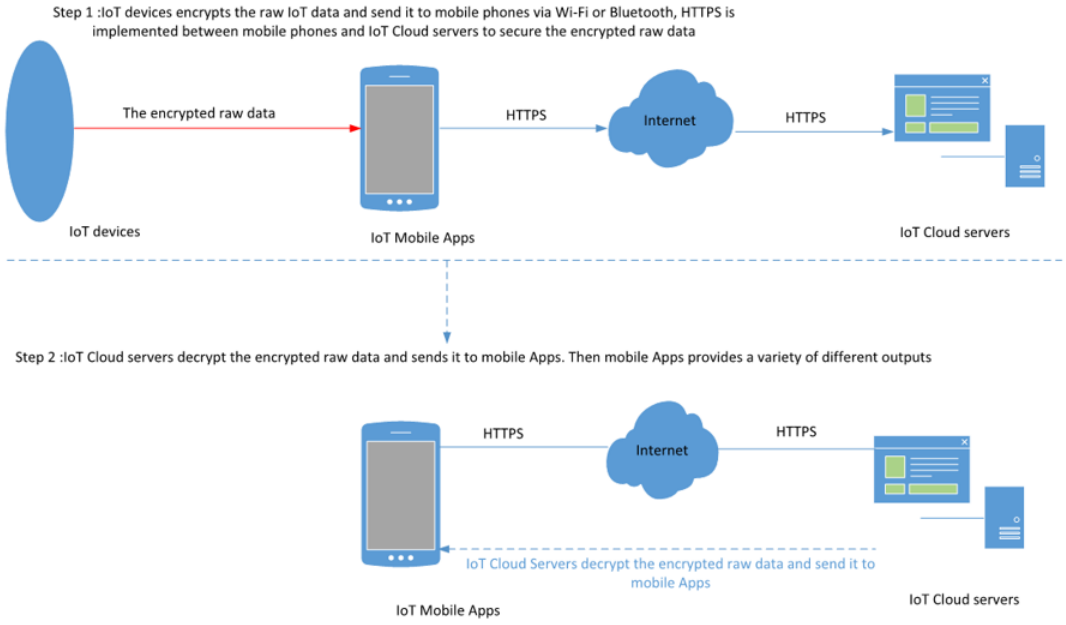

Figure 6. Our recommended security architecture for IoT Devices

A temporary solution is to avoid sending the plaintext request data and the response data or reduce the frequency of uploading plaintext data. This countermeasure is a temporary measure before other encryption algorithms

replace the Base 64 usage. The long-term solution is still to encrypt the request and response data between the mobile and Fitbit server because most of the flows involve an authorization credential. In the tests, the traffic containing information about Fitbit app, uploaded the serial number of Tracker and the Fitbit server openly. It also returned "authSubkey" and "nonce" (July 2018). In August, this flow was rarely captured. It is possible that the vendor found out about this potential risk, and upgraded their Fitbit application software. One solution for Fitbit is to reduce the frequency of uploading the user's mobile information. There is no need to collect the user's mobile phone information every time when the user synchronizes the Fitbit application and the Fitbit server. Moreover, Fitbit can cancel the response data of the user profile because the profile is not required to send back to the user and the Fitbit application. Instead, the data may be intercepted by malicious people who can sell the private data for profit. Figure 6 provides our recommended architecture for IoT security that reflects the learning gained in the experimentation with the Fitbit Information System.

CONCLUSION

Communication security risk can come from heterogeneous network connections and network service vulnerabilities. IoT networks involve cross-connection between different networks. For instance, ZigBee, RFID, Bluetooth, wireless LAN, and 4G/5G, ethernet. These different types of networks involve different protocols. This makes it more difficult to protect IoT data confidentiality, integrity, and availability. Network service also has its vulnerabilities, for example, TCP/IP network, Bluetooth, and wireless have their own security weaknesses. HTTP is proved insecure. Bluetooth can be hijacked. Wireless threats can come from malicious association, mac spoofing, MITM, and ad-hoc network intrusion. Service risk includes the risks of local services, the risks of IoT manufacturers', Cloud services, and the risks of the third-party Cloud services. IoT devices have their own access or login methods, but these methods have weakness in authentication. For example, the web login of these devices only supports HTTP and not HTTPS, or the user only sets a simple password. Although the user sets one strong password, an IoT device may not provide a mechanism to limit or reduce the number of password attempts. IoT manufactures usually use the Cloud to address this challenge of huge amounts of user data storage. But the problem is how to ensure the privacy and security of user data, and how to know if IoT vendors sell the data outside for profit. This research has explored security vulnerabilities in one IoT device and it's Information System. Suggested countermeasures and security solutions are proposed that can be implemented more generally to address weaknesses in the current IoT architectures.

REFERENCES

Amadeo, M., Campolo, C., Quevedo, J., Corujo, D., Molinaro, A., Iera, A.,Vasilakos, A. V. (2016). "Information-centric networking for the internet of things: challenges and opportunities". *IEEE Network*, 30(2), 92-100. https://doi.org/10.1109/MNET.2016.7437030.

Classen, J., Wegemer, D., Patras, P., Spink, T., & Hollick, M. (2018). "Anatomy of a Vulnerable Fitness Tracking System: Dissecting the Fitbit Cloud, App, and Firmware". *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2(1), 5. https://doi.org/10.1145/3191737

Cyr, B., Horn, W., Miao, D., & Specter, M. (2014). "Security analysis of wearable fitness devices (fitbit)". Retrieved from https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2015/03/20082016/17-cyrbritt-webbhorn-specter-dmiao-testing-fitbit.pdf.

Farooq, M. U., Waseem, M., Khairi, A., & Mazhar, S. (2015). "A critical analysis on the security concerns of internet of things (IoT)". *International Journal of Computer Applications*, 111(7). Retrieved from http://www.pcporoje.com/filedata/592496.pdf

Fereidooni, H., Classen, J., Spink, T., Patras, P., Miettinen, M., Sadeghi, A.-R., Conti, M. (2017). "Breaking fitness records without moving: Reverse engineering and spoofing fitbit". Springer. Proceeding of 20th International Symposium on Research in Attacks, Intrusions, and Defenses. Atlanta, GA. https://doi.org/10.1007/978-3-319-66332-6_3

Fereidooni, H., Frassetto, T., Miettinen, M., Sadeghi, A.-R., & Conti, M. (2017). "Fitness Trackers: Fit for health but unfit for security and privacy". Proceedings of the 2017 IEEE/ACM International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE). Philadelphia, PA. https://doi.org/10.1109/CHASE.2017.54

Gaur, P., & Tahiliani, M. P. (2015). “Operating Systems for IoT Devices: A Critical Survey”. Paper presented at 2015 IEEE Region 10 Symposium, 33-36. Ahmedabad, India. https://doi.org/10.1109/tensymp.2015.17

Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). “Internet of Things (IoT): A vision, architectural elements, and future directions”. *Future Generation Computer Systems*, 29(7), 1645-1660. https://doi.org/10.1016/j.future.2013.01.010

Mahmoud, R., Yousuf, T., Aloul, F., & Zualkernan, I. (2015). “Internet of things (IoT) security: Current status, challenges and prospective measures”. Proceedings of the 10th International Conference for Internet Technology and Secured Transactions (ICITST), London, UK. https://doi.org/10.1109/ICITST.2015.7412116

Mendes, T., Godina, R., Rodrigues, E., Matias, J., & Catalão, J. (2015). “Smart home communication technologies and applications: Wireless protocol assessment for home area network resources”. *Energies*, 8(7), 7279-7311. https://doi.org/10.3390/en8077279

Rahman, A. F. A., Daud, M., & Mohamad, M. Z. (2016). “Securing Sensor to Cloud Ecosystem using Internet of Things (IoT) Security Framework”. Proceedings of the International Conference on Internet of things and Cloud Computing. Cambridge, United Kingdom. https://doi.org/10.1145/2896387.2906198

Tilkov, S. (2015). “The Modern Cloud-Based Platform”. *IEEE Software*, 32(2), 116-116. https://doi.org/10.1109/ms.2015.51

Trappe, W., Howard, R., & Moore, R. S. (2015). “Low-energy security: Limits and opportunities in the internet of things”. *IEEE Security & Privacy*, 13(1), 14-21. https://doi.org/10.1109/MSP.2015.7

Wu, M., Lu, T.-J., Ling, F.-Y., Sun, J., & Du, H.-Y. (2010). “Research on the architecture of Internet of Things”. Proceedings of the 2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE), Chengdu, China. https://doi.org/10.1109/ICACTE.2010.5579493