



Article

Enhancing Privacy in Wearable IoT through a Provenance Architecture

Richard K. Lomotey ^{1,*} , Kenneth Sofranko ¹ and Rita Orji ²

¹ Department of Information Sciences and Technology (IST), Pennsylvania State University, Beaver Campus, Monaca, PA 15061, USA; kms745@psu.edu

² Faculty of Computer Science, Dalhousie University, Halifax, NS B3H 4R2, Canada; rita.orji@dal.ca

* Correspondence: rkl5137@psu.edu

Received: 23 February 2018; Accepted: 20 April 2018; Published: 23 April 2018



Abstract: The Internet of Things (IoT) is inspired by network interconnectedness of humans, objects, and cloud services to facilitate new use cases and new business models across multiple enterprise domains including healthcare. This creates the need for continuous data streaming in IoT architectures which are mainly designed following the broadcast model. The model facilitates IoT devices to sense and deliver information to other nodes (e.g., cloud, physical objects, etc.) that are interested in the information. However, this is a recipe for privacy breaches since sensitive data, such as personal vitals from wearables, can be delivered to undesired sniffing nodes. In order to protect users' privacy and manufacturers' IP, as well as detecting and blocking malicious activity, this research paper proposes privacy-oriented IoT architecture following the provenance technique. This ensures that the IoT data will only be delivered to the nodes that subscribe to receive the information. Using the provenance technique to ensure high transparency, the work is able to provide trace routes for digital audit trail. Several empirical evaluations are conducted in a real-world wearable IoT ecosystem to prove the superiority of the proposed work.

Keywords: Internet of Things (IoT), sensors; mobile devices; middleware; wearables; provenance; privacy

1. Introduction

There are several billions of smart, connected “things” around us today that enable the creation of new opportunities, use cases, and applications [1–4]. Primarily, these devices are equipped to collect heterogeneous types of data from various areas such as homes and cities to vehicles and roads to tracking objects for an individual's behavior; and the purpose is to push the collected data to cloud services for analytics.

There are several use cases and applications of IoT and especially in healthcare [5], the list includes: Telehealth: remote or real-time pervasive monitoring of patients, diagnosis and drug delivery. With wearable IoT for fitness tracking, sensors are able to read users' vitals and the information can be pushed to healthcare facilities.

However, the hunger for sensor data streaming in an attempt to automate clinical data exchanges for instance, can create the environment for privacy breaches. Taking wearable IoT for example, smartwatches and other sensors (e.g., blood oxygen reader, gamma ray radiation detectors, etc.) are facilitated to collect personal records including vitals, location, dosage, and so on. These devices are constantly running and ready to send data anytime a requesting node makes a demand. This means users mostly put the devices in broadcast mode so that they are easily discoverable for the data exchange process. Once these wearable devices are within discoverable range, any other node can make a request and if proper privacy measures are not in place, personalized data can be stolen by unauthorized persons.

Recently, some IoT device manufacturers, such as Texas Instrument Inc., have built-in state-of-the-art hardware security technology incorporated into their devices. However, this does not guarantee much user protection if the IoT environment follows the broadcast model.

Hence, by focusing on wearable IoT and personal data exchanges, this paper proposes the broadcast-subscriber IoT model where users' personal data are only shared with intended nodes such as healthcare facilities or devices authorized by the user. This is achieved in two major folds.

Firstly, we propose a meta-data level encryption techniques where the identifiable components of the communicating devices (e.g., serial number, MAC address) are encrypted. The encryption-decryption keys are only known to the subscription devices and each time newly generated encrypted data is delivered to a subscriber, the latter can only decrypt the data if it belongs to the subscription pool because that is the only way the decryption key which is prior shared can be known. Several encryption-decryption algorithms at both the hardware and software levels are tested including the AES, DES, 3DES, MD5, and SHA.

Secondly, the provenance technique is proposed in the broadcast-subscriber IoT architecture to ensure transparency and full digital audit. In most instances when privacy is breached (e.g., personal data stolen) in wearable IoT, it is difficult to detect the breach since devices are not able to maintain proper historical records of previous contacts. With the proposed provenance technique, the paper pushes for data trace route detection and formulation which is presented to the user through visualization techniques. With provenance, the personal data is tracked from the point of generation to the current state.

The *proposed broadcast-subscriber IoT model* is a variation of a *multicast-based model* except in the former, it is mandatory that you register/subscribe prior to the message being broadcast. The drawback in the general multicast approach is any device within a discoverable region can access the IoT data whereas our approach will not allow that if you are not registered prior to the message being published/broadcast.

Overall, the work made the following contributions to the personal and ubiquitous computing research:

- Provenance is proposed to generate data trace routes in the wearable IoT economy order to ensure digital audit trail for transparency. The work adopted the broadcast-subscriber IoT architecture to ensure privacy of users' data such as vitals especially in wearable IoT; a key concern in the generic broadcast IoT architecture.
- Both the hardware and software level data encoding methodologies are proposed based on device meta-data encryption.

The remaining sections of the paper are arranged as follows. Section 2 underscores the importance of IoT in the current dispensation as well as reviews of works on IoT and privacy. Section 3 describes the architectural design of our proposed broadcast-subscriber IoT model and design justifications while the evaluation of the implementation is carried out in Section 4. The paper concludes in Section 5 with our contributions and future research direction.

2. Background Works

2.1. IoT Applications

With the Internet of Things (IoT), addressable machines (or objects) such as smart sensors and other physical devices that are ideally not seen as computers, can interact with less human intervention [1,6–9]. This has given rise to new use cases in the management and sharing of personal data with caregivers where sensors and smartwatches can stream personal vitals; an area known as wearable IoT. This is commonly seen in fitness and health monitoring applications.

However, there are several communication protocols and standards (e.g., NFC, RFID, Bluetooth, Bluetooth LE, ANT, Proprietary, Wi-Fi, ZigBee, Z-wave, 6LoWPAN, 2.5–4 G, etc.) which can lead to most of the challenges including limited interoperability [1]. Moreover, compromising data is cheaper

because there are several protocols that can be explored on the same device for digital content accessibility. Hence, in wearable IoT for instance, users who are streaming their personal vitals to backend services (e.g., health information systems in the cloud) or across multiple personal devices (e.g., streaming between sensors and smartphones for fitness tracking) are at risk from hackers. This case is even rampant when the broadcast technique is employed in the wearable IoT architecture because then any device within discoverable range can attempt to retrieve information.

These challenges with personal data compromises creates the need for data encoding and the generation of audit trails to determine how the IoT data is shared between devices. This is why this work further reviewed some works on privacy-based provenance systems.

2.2. Privacy and Provenance in IoT

Over the years, provenance (i.e., the record of the lineage of data) has proven to ensure quality, reliability and to a very large extent transparency, in enterprise systems. In the Electronic Health System, data provenance tracking is necessary for rights protection, regulatory compliance, management of intelligence and medical data, and verifying of information as it flows through various stages [10]. Moreover, as huge amounts of data are being generated, provenance may be required to ensure privacy [11,12].

Recently, the research community has started looking into the employment of provenance to answer the what, when, and how questions on IoT data. Eduardo et al. [13] proposed a lightweight semantic model and a prototypic mobile-enabled software for capturing information about IoT devices, including their provenance, capabilities and use so as to enable users make inferences about them. According to the authors, using provenance provides the understanding of the lifecycle of data and the reason for which it was used in the IoT system.

On the other hand, Jan et al. [14] provide an in-depth analysis of privacy threats and challenges in the Internet of Things which brings new issues such as pervasive privacy-aware management of personal data. The authors named seven categories of privacy violations which include identification, tracking and profiling, threats of privacy-violating interactions and presentations, lifecycle transitions, inventory attacks and information linkage.

Similarly, Elisa et al. [15], noted that privacy is very vital in the context of IoT since data collection includes context data such as location, time, etc., which enables inferences on personal information and preferences of individuals. Thus, such information needs protection by all parties involved from its capturing, management, and its use, to its storage [16]. However, little has been done regarding security, privacy and personal safety risks that arise beyond subsystems; that is, cross platform openness perceptible that comes with cloud services in relation to IoT [17].

Within the context of healthcare, emphasis on maintaining originality of which data provenance is required, in medical applications is key [18]. This can prevent passive attacks like eavesdropping of contextual information and spoofing attacks. Further, the IoT can be useful for hospitalized patients whose physiological status requires close attention by adopting noninvasive monitoring.

2.3. The Open Issues

When one takes a closer look at the literature reviewed, it is apparent that there is not much work done on securing personal health data (e.g., vitals) in IoT systems. This can be attributed to the fact that the field is relatively new. Meanwhile, most of the works reviewed on privacy issues in IoT do not have concrete architectural designs. Besides, while techniques such as provenance have been successful in other domains for privacy purposes, the technique is under-utilized in IoT.

In the wearable IoT, streaming and sharing personal and contextual data across devices require even higher level of privacy. Unfortunately, just broadcasting information from the IoT devices does not help because sniffers can be attracted. Thus, this paper has put forward the following research questions to explore specifically to guarantee privacy when sharing personal data:

1. How can the personal data shared in wearable IoT be made more secure when devices are broadcasting?
2. How can the data be encoded in a way that even if the shared keys are stolen, data privacy will not be breached?
3. Can transparency be offered to ensure the determination of who got what information in the wearable IoT?

These questions are addressed in the upcoming sections.

3. The Designed IoT Architecture

3.1. Designing the System

Based on the aforementioned challenges, an IoT architecture is proposed and developed in [19]. The diagram of the architecture as reproduced is shown in Figure 1. The design of the architecture is to facilitate secure IoT data transmission in a multi-tier network that comprises: IoT devices, middleware, and some cloud-hosted applications and services. Though our focus is on wearable IoT devices, the architecture is generic enough to support interactions with devices in mainstream IoT services such as: Home, Work, and Vehicular IoT. To facilitate this, the work explores provenance and broadcast-subscriber scenarios.

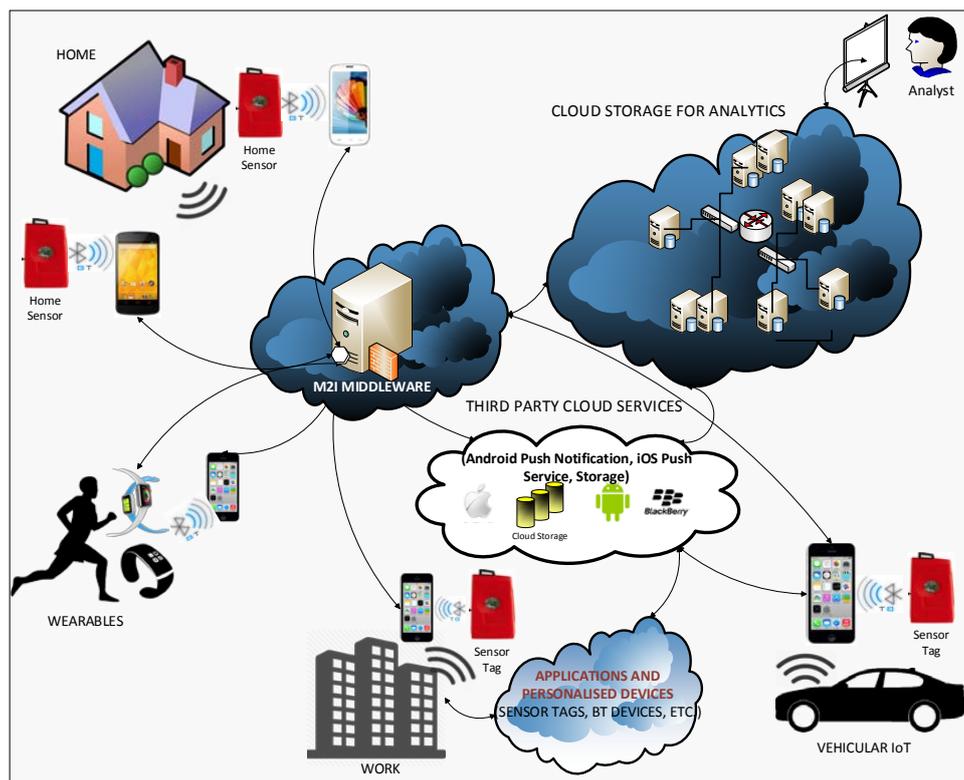


Figure 1. The generic IoT architecture showing major segments where interactions can occur.

This is the reason the overall architecture has cloud-hosted backend services and the client IoT devices. The proposed middleware enables data collection for provenance tracking and can also act as a centralized authority. The *Push Notification* component comprises third party cloud services from Google, Apple, and BlackBerry that allow us to send messages to a mobile notification center. These types of push information are consumable by both end-users on their mobile devices and the machines especially in decision making scenarios. The analytics layer is where the final composed

data is stored in a well formatted fashion. This data can then be used by the analysts to perform the audit trail which is offered through the provenance data.

With regards to the types of devices employed for testing, the following devices are considered in this work.

Limited range that communicate primarily via short span protocols such as Bluetooth, Bluetooth Low Energy (BLE), Near-field communication (NFC), and Radio-frequency identification (RFID). Example is shown in Figure 2A, specifically, the CC2650 SensorTag [20] manufactured by Texas Instruments. In Figure 2B,C the various readings from the device during pairing via a BLE on a tablet and smartphones respectively are shown.

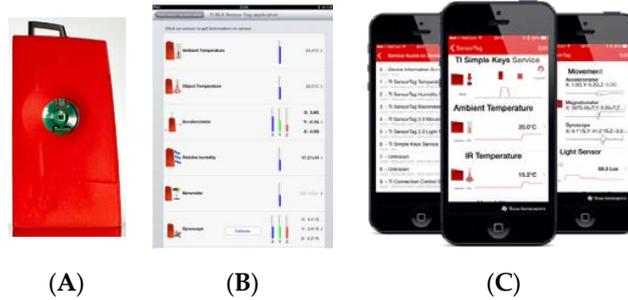


Figure 2. (A) Sample SensorTag. (B) Sample application on iPad. (C) Sample app on mobile devices showing readings.

Another smart device engaged in the work is the Optical Heart Rate Monitor with BLE. Likewise, the Raspberry Pi 3 with an integrated 802.11n wireless LAN and Bluetooth employs a similar functional architecture.

Mobile devices (e.g., smartphones, tablets, and smartwatches) as the main access point of data between the sensor devices and the cloud services.

In the implementation, the Xamarin [21] development tool is used that guarantees programming in C#. This enables the development of a native app that integrates with the sensor API.

The storage facility employed is CouchDB [22]. A sample IoT data stored in the CouchDB storage is shown in Figure 3.

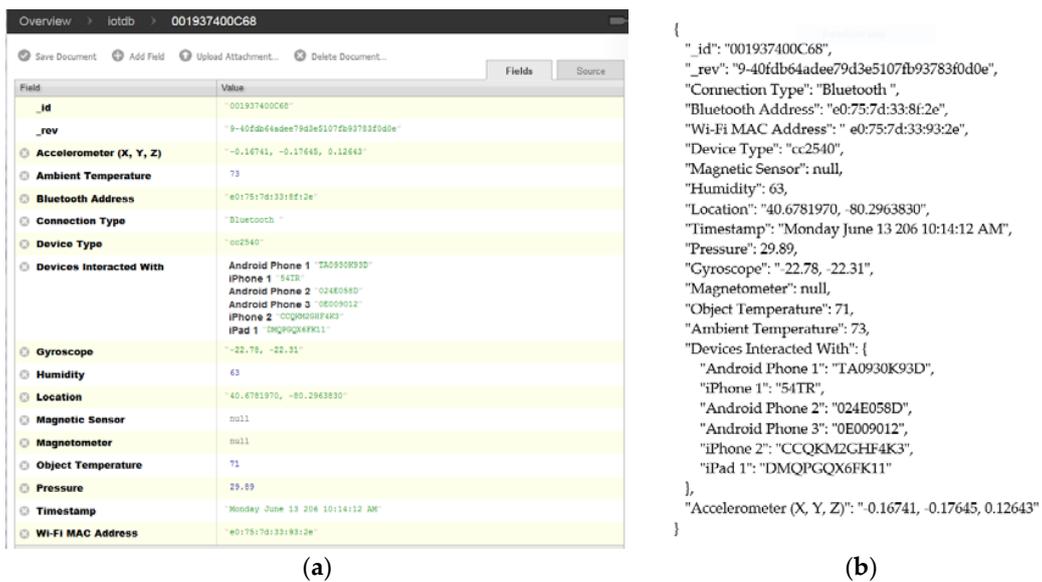


Figure 3. Sample IoT provenance data stored in our CouchDB database—Fields View (a) and Source View (b).

The architecture is designed so that the wearable IoT devices can exchange data with the back-end component. The shared services segment is a pure data-centric exchange medium which has been designed as a typical data distribution service; meaning, the IoT data delivery is based on subscriptions where the users will have to register their devices in a subscription pool to which dissemination can take place. Our proposed system can also be integrated into other gateways such as Kura [23].

3.2. Provenance and Audit Trail

Now that the entire data flow process in the wearable IoT architecture is explained, we can tell the various levels where data can be stolen and consequently breach privacy. Even though hardware and software level encodings are proposed, sniffing and masking attacks can still happen. Also, there are several communication interfaces that can be exploited. Moreover, just proposing a broadcast-subscriber mechanism is not enough as attackers can attempt to enter the subscription pool. Thus, we proposed device level meta-data encryption first before offering provenance for digital auditing.

This means devices will have to share unique features (such as device UUID, MAC address, etc.) which will be used as part of the generated key. In this regard, any device which will be making a request with say an unidentified UUID will not be honored. The main concern that arises is how the system is monitored to ensure that unauthorized access is prevented to ensure privacy. This is where provenance is proposed in the following phases. Our proposed provenance approach is adapted from [24].

Scheme Preparation: In this phase, the middleware does the instantiation of the required configuration. Considering a security parameter k , the middleware initially creates the bilinear parameter $(p, g_1, g_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ by running $\mathcal{Gen}(k)$, and chooses two secure cryptographic hash function H, H_1 where $H : \{0, 1\}^* \rightarrow \mathbb{G}_1$ and $H_1 : \{0, 1\}^* \rightarrow \mathbb{Z}_p^*$. Then, the middleware chooses an element $h \in \mathbb{G}_1$, and $\xi_1, \xi_2 \in \mathbb{Z}_p^*$, and sets $u, v \in \mathbb{G}_1$ such that $u^{\xi_1} = v^{\xi_2} = h$. With these settings, the middleware keeps the master keys (ξ_1, ξ_2) secretly, and publishes the public parameters $Pubs = (p, g_1, g_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, h, u, v)$.

The middleware, say $TS_j \in \{TS1, TS2, \dots\}$ chooses a random number $\gamma_j \in \mathbb{Z}_p^*$ as its private key, and publishes the corresponding public key $w_j = g_2^{\gamma_j}$. These random numbers can be based on the unique identifiers of the IoT devices.

When an IoT device $U_i \in \mathcal{U}$ is registered to the middleware TS_j , TS_j chooses a random number $x_i \in \mathbb{Z}_p^*$ such that $\gamma_j + x_i \neq 0$ and computes $A_{ji} = g_1^{\frac{1}{\gamma_j + x_i}}$, and returns (x_{ji}, A_{ji}) as the private key of U_i . In addition, the middleware keeps (U_i, A_{ji}) in a list for the provenance monitoring later.

Provenance Record Creation: The data provenance generation for a single document's creation, write and read operations in a middleware TS_1 , where the private-public key pair of TS_1 is $(\gamma_1, w_1 = g_2^{\gamma_1})$ is explained based on the document creation and monitoring steps.

Document Creation: When an IoT device U_i wants to log into the middleware TS_1 to broadcast data (which requires write operation), the following steps has to be followed first for the purpose of authentication:

Step 1: Choose some random numbers $\alpha, \beta, r_\alpha, r_\beta, r_{x_i}, r_{\delta_1}, r_{\delta_2} \in \mathbb{Z}_p^*$, and compute

$$\left\{ \begin{array}{l} T_1 \leftarrow u^\alpha, T_2 \leftarrow v^\beta, T_3 \leftarrow A_{1i} h^{\alpha+\beta} \\ \delta_1 \leftarrow x_{1i} \alpha, \delta_2 \leftarrow x_{1i} \beta, R_1 \leftarrow u^{r_\alpha}, R_2 \leftarrow v^{r_\beta} \\ R_3 \leftarrow e(T_3, g_2)^{r_{x_i}} \cdot e(h, w_1)^{-r_\alpha - r_\beta} \cdot e(h, g_2)^{-r_{\delta_1} - r_{\delta_2}} \\ R_4 \leftarrow T_1^{r_{x_i}} \cdot u^{-r_{\delta_1}}, R_5 \leftarrow T_2^{r_{x_i}} \cdot v^{-r_{\delta_2}} \end{array} \right.$$

Step 2: Choose the current time stamp CT and compute the challenge c , where

$$C \leftarrow H_1(CTT_1T_2T_3R_1R_2D_3R_4R_5)$$

Step 3: Compute

$$\left\{ \begin{array}{l} s_\alpha = r_\alpha + C\alpha, s_\beta = r_\beta + C\beta \\ s_x = r_{x_i} + Cx_{1i}, s_{\delta_1} = r_{\delta_1} + C\delta_1, s_{\delta_2} = r_{\delta_2} + C\delta_2 \end{array} \right.$$

Step 4: Send the timestamp CT and $\sigma = (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_{\delta_1}, s_{\delta_2})$ to the middleware for authentication.

Upon receiving CT and σ , the middleware TS_1 first checks the timestamp CT to avoid replay attack. Then, the middleware runs the following steps to verify $\sigma = (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_{\delta_1}, s_{\delta_2})$:

Step 1: Compute R_1, R_2, R_3, R_4 and R_5 :

$$\left\{ \begin{array}{l} \tilde{R}_1 \leftarrow u^{s_\alpha} / T_1^c, \tilde{R}_2 \leftarrow v^{s_\beta} / T_2^c \\ \tilde{R}_4 \leftarrow T_1^{s_x} / u^{s_{\delta_1}}, \tilde{R}_5 \leftarrow T_2^{s_x} / v^{s_{\delta_2}} \\ \tilde{R}_3 \leftarrow e(T_3, g_2)^{s_z} e(h, w_1)^{-s_\alpha - s_\beta} e(h, g_2)^{-s_{\delta_1} - s_{\delta_2}} \\ \quad (e(T_3, w_1) / e(g_1, g_2))^c \end{array} \right.$$

Step 2: Check the following equation

$$c \stackrel{?}{=} H_1(CT || T_1 || T_2 || T_3 || \tilde{R}_1 || \tilde{R}_2 || \tilde{R}_3 || \tilde{R}_4 || \tilde{R}_5)$$

If it holds, the middleware TS_1 can confirm that the request is from a subscribed IoT. Otherwise, the middleware TS_1 rejects the attempt to acquire the data. Once a device is authenticated, that device say U_i can create a new document M , and store it in the middleware TS_1 . Then, the middleware TS_1 generates an authenticated provenance for the document creation operation as follows:

$$\sigma_1 = H(h_1 M_v P_1)^{\gamma^1},$$

with

$$h = \text{Create}''$$

where “Create” represents the document creation task, M_{v_1} indicates version-1 of the document M , and $P_1 = CT || T_1 || T_2 || T_3$ is the provenance.

Data Provenance Monitoring: Suppose the created document under consideration is in version j in a late time, the middleware and the IoT device which is the originator of the data can pool resources to track the identity of the IoT device which created this version. This requires the following steps to be followed:

Step 1: The IoT device or the middleware first takes a version of the provenance record $[h_i || M_{V_i} || P_i || \sigma_1]$ and sends the provenance information $P_j = CT || T_1 || T_2 || T_3$ to an independent trusted authority, which is part of the middleware but a detached component.

Step 2: The trusted authority, say TA uses its master key (ξ_1, ξ_2) to compute and based on the outcome return $A \leftarrow T_3 / (T_1^{\xi_1} \cdot T_2^{\xi_2})$ to the middleware TS_1 .

Step 3: The middleware TS_1 then looks up the tracking list with A . If an entry (U_i, A_{1i}) is found with $A_{1i} = A$, the IoT device U_i which created the questionable version is tracked. The correctness is that, if the version is really generated by U_i , we will have

$$A = \frac{T_3}{(T_1^{\xi_1} \cdot T_2^{\xi_2})} = A_{1i} \cdot \frac{h^{\alpha+\beta}}{u^{\alpha\xi_1} \cdot v^{\beta\xi_2}} = A_{1i} \cdot h^{\alpha+\beta} / (h^\alpha \cdot h^\beta) = A_{1i}$$

The proposed provenance methodology is able to achieve user privacy preservation in a wearable IoT system. This is because when an IoT device U_i subscribes into the middleware TS_1 , it provides the anonymous authentication information (CT, σ) , and maintains user identity privacy and the login unlinkability of the same device. Moreover, when an IoT device creates a questionable document or document version, the collaboration between the middleware and the other devices can track the real identity of a particular requesting node within the subscription pool.

Also, the proposed provenance methodology can offer user privacy disclosure independence within the wearable IoT system. The use of the device meta-data as part of the formation of the random public and private key generations will be difficult to duplicate. Even if hackers are able to guess a unique device information say the UUID, the fact that the number is randomly generated and the UUID is not used directly will deter attacks.

4. Evaluation

In this section, empirical evaluations are provided to validate the proposed privacy enhanced wearable IoT architecture. Four separate evaluations are conducted to determine: (1) the device resource utilization cost, (2) the data propagation speed for different communication protocols, (3) the robustness of the provenance-oriented IoT system, and visual analytics of the data trace routes.

The BLE *SensorTag* is used as the sensor nodes in conducting the evaluations due to its support for different network protocols as well as varied data generation. Some of the quality of service (QoS) properties under consideration in the work are provided in Table 1.

Table 1. QoS properties.

Property	Definition
Resource Utilization	The percentage of resources taken by the encryption schemes.
Network Types	Different communication protocols such as ZigBee, HTTP, Bluetooth, and BLE.
Encryption/Decryption Schemes	AES (aes_128_cfb), DES (des_cfb), 3DES, MD5, and SHA (SHA-256).
Hashing Schemes	MD5, and SHA (SHA-256).
Soft-Real Time	Acceptable window within which data should be synchronized.
Fault Injection	Attempts to hack the proposed broadcast-subscriber system for the purpose of testing.

4.1. The IoT Device Resource Utilization Cost

Proposing privacy mechanisms in an IoT architecture must first of all be grounded on the feasibility of the device to handle the required processing load. We considered encryption algorithms such as: AES, DES, 3DES and hashing algorithms such as: MD5 and SHA. We categorize the utilization of the CPU and RAM based on the various encryption and hashing algorithms. The security components of the application (described in Table 1) is installed on the SensorTag and the utilization is observed. The recorded data is the candidate dataset under consideration and plotted in Figure 4.

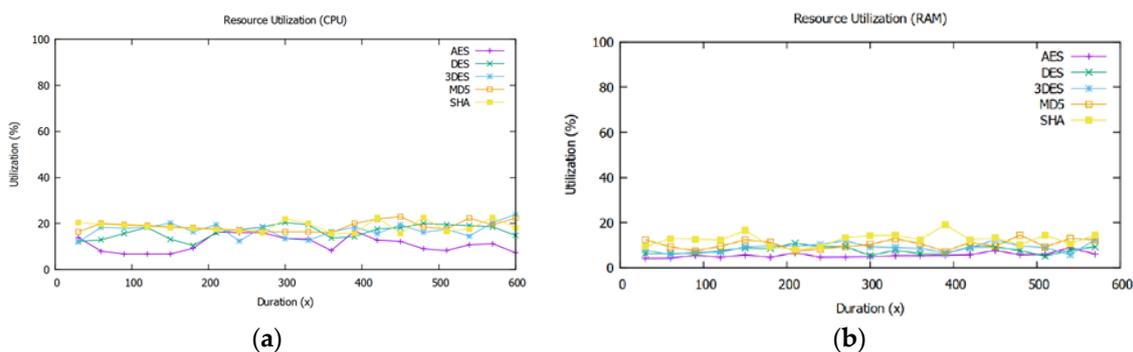


Figure 4. The IoT device utilization of the middleware (a) CPU (L) and (b) RAM (R).

The hashing algorithms are employed specifically for integrity check especially when the data is corrupted. In that case, the sending device and the receiving devices can use the hash values as checkpoints. The encryption techniques however, are proposed as a data protection measure. The data plotted represents the overall utilization for both the encryption/decryption processes as well as the hashing techniques. This experiment is repeated several times with five (5) different sensors and four (4) smartphones. A summarized view of the experiment is presented in Table 2, highlighting

the minimum, average and maximum utilization points within the test duration. Starting from 30 s, the utilization points are recorded until 600 s for the different device resources.

Table 2. Summarized utilization result.

Resource		Utilization		
		Min (%)	Avg (%)	Max (%)
CPU	AES	6.75	11.15	16.75
	DES	10.44	16.51	20.44
	3DES	11.89	17.10	23.96
	MD5	16.02	18.84	23.00
	SHA	15.65	18.68	22.66
RAM	AES	4.25	5.66	8.96
	DES	5.23	7.82	10.95
	3DES	5.87	9.00	12.44
	MD5	7.33	10.50	14.56
	SHA	8.22	12.67	19.22

Clearly, we can see that the average device processor utilization of the device is approximately 16% which is really encouraging because it means that majority of the CPU is dedicated to other activities and the user experience will not be affected. This is similar for the memory (RAM) with average utilization of 9%. Regarding the variation of the utilization points across the various algorithms, the difference is not much noticeable even though some have lower utilization than others. It must be pointed out that our aim is not to validate the superiority of one algorithm over the other, but rather, how each supports the privacy enhancing process of the proposed system. However, the result is worth discussing. When no crypto module is in use (i.e., the encryption/decryption and hashing), the maximum CPU usage is around 9.41% and the maximum RAM usage is around 6%.

The AES shows much lower utilization points compared to the other encryption/decryption algorithms. This phenomenon is consistent across the results for both the processor and memory consumptions. This result is encouraging for the proposed system to be adopted.

4.2. Sensor Data Propagation

In the IoT environment, one of the key requirements is soft-real time data propagation. Due to the heterogeneity in a M2M communication, the analysis here focuses on the latency implications of four (4) protocols namely: Bluetooth, Bluetooth Low Energy (BLE), HTTP, and ZigBee. The evaluation took into account the data propagation speed between the IoT devices (e.g., sensors) including the generation and monitoring of the provenance records. The results are plotted in Figure 5. This evaluation is considered in two broad spectrums, (1) the speed with which the broadcast-subscribe scheme identifies the type of network required for the M2M communication and (2) the cost of detecting an input and output (I/O) interface for the data sharing process. In the design of the broadcast-subscriber system, we developed 40 controller classes which we call the application interfaces. The number of interfaces that can be activated is based on how many M2M communications and the provenance records a user wants to track.

In the first setup, we considered the network type detection cost. The average propagation speed is 13.52 ms for the Bluetooth, 11.55 ms for the BLE, 12.84 ms for the HTTP, and 20.81 ms for ZigBee. Overall, we observed that the number of application interfaces has no direct impact on the propagation speed. This observation is consistent across the various M2M protocols. This is because regardless of the number of application interfaces, it takes the same effort to identify the network type. However, it is observed that some of the communication protocols have significantly higher latency than others. An example is say the Bluetooth has lower latency compared to ZigBee. One element that plays into the differentiation is the range between the devices.

The second setup is the cost of detecting an I/O interface. This evaluation is important because we need to study the rate at which an IoT device is able to send a request and also respond to an incoming request. This experiment however has a linear increment in time for increasing number of

interfaces. For instance, the average time for processing at the I/O level is 16.13 ms for the Bluetooth, 26.75 ms for BLE, 38.10 ms for HTTP, and 78.43 ms for ZigBee. While these statistical averages are encouraging, a study of the various number of application interfaces shows a linear rise. This is because, every request has to be processed at the I/O level so smaller number of application interfaces will have faster times compared to larger number of application interfaces.

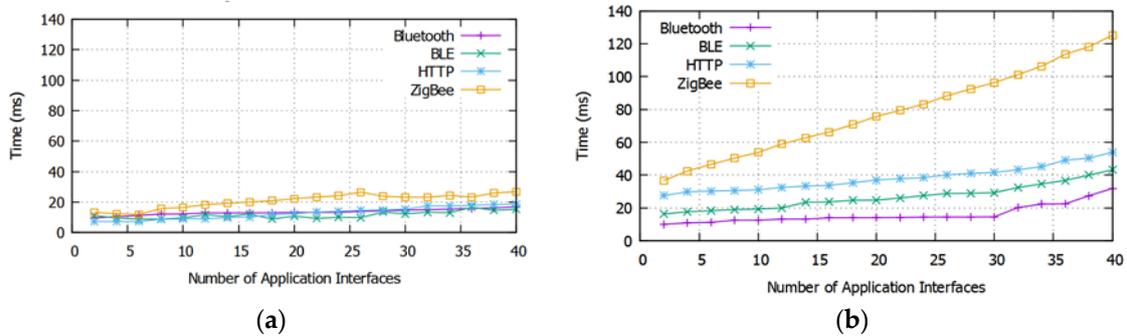


Figure 5. (a) The average round trip of network type detection, (b) The average I/O processing time

4.3. Fault Injection Analysis

The fault injection analysis is conducted to evaluate the resilience of the proposed privacy-aware broadcast-subscriber IoT wearable architecture. This evaluation is conducted by the research team together with user recruits from the Pennsylvania State University. The test group involves 30 user devices and the worst cases of system compromises that can lead to personal data theft are considered. Especially, the *mask attack*, which is a variation of the Brute-force attack is considered. In the Brute-force attack, a password guess work is required in an attempt to penetrate a system. However, with mask attack, the assumption is some characters of the password are known so it reduces the number of times verification is required. This attack is considered more dangerous and realistic in situations where theft can occur through persons or systems that have prior knowledge about the user.

In conducting the evaluation, users are required to use our broadcast-subscriber IoT system only to share their personal vital through activity tracking. The users agreed to give us their passwords which they used for the device-level meta-data encryption. This means for the mask attack to succeed, the device feature such as the UUID is what must be verified. This is because as posited, the encryption and decryption of the data is done using keys that are formulated based on user passwords and device features in the subscription pool. So basically, the fault injection analysis focuses on attacks within the subscription pool between registered devices, a more severe level of data breaches if there is ever a comprise of the system. The evaluation results show high level of security resilience as the true positive and false positive results are plotted in Figure 6. In the same figure, the x-axis shows the amount of data block, measured in KB, considered for each round of testing. The data block represents the size of the encryption keys and the amount of IoT data being propagated by the various devices. The variation in size is determined by how many devices are actively engaged simultaneously.

The true positive results represent the scenarios where the personal vitals is decrypted either correctly by an authorized subscriber or genuinely, the decryption is unsuccessful because an attack is detected. Overall, the true positive is encouraging considering the closeness of the attacks we carried including guessing the passwords and, in some cases, having access to some device UUIDs. Most of the encryption-decryption algorithms (i.e., AES, DES, 3DES, MD5, and SHA) show high true positive outcomes in the mid 90%. Our experiments however show that some of the algorithms can be more superior such as the 3DES.

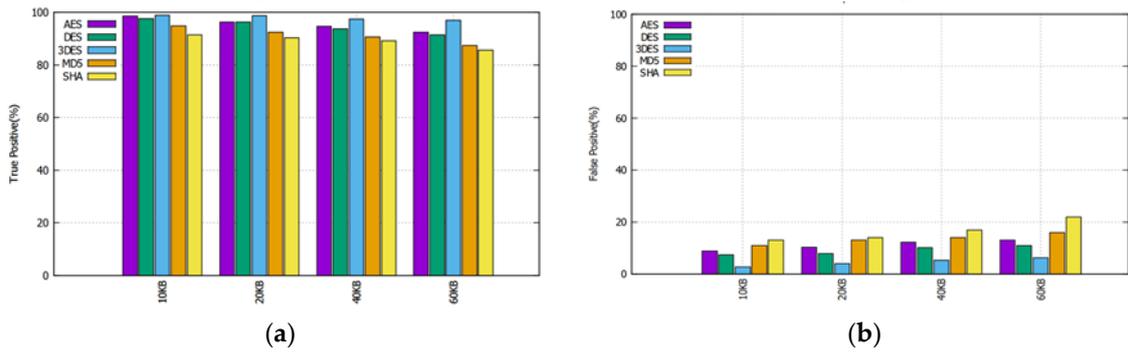


Figure 6. The fault injection analysis showing the (a) true positive (L) and (b) false positive (R).

The false positive results represent scenarios where a thief is able to decrypt a data or a genuine registered device within the subscription pool is misclassified as an attacker and subsequently denied access. This misclassification is the main result plotted in Figure 6 as the case of thieves decrypting the data is recorded to be almost zero percent. It is observed that the reason for the misclassification in most cases is because the wearable devices did not communicate their UUIDs but other features which are unstable such as Bluetooth address. Some devices also communicate their MAC addresses which in some instances will not match the UUID.

The high false positive in some of the algorithms as seen in the plot is also due to the processing speed variations in the schemes. So, in terms of resilience, they may have marginal differences but in terms of efficiency, clearly some are more superior.

4.4. Visual Analytics from the Provenance Data

This evaluation is an extension of the fault injection. Here, we focus on the provenance data which is stored to investigate the interconnectedness between the IoT devices by composing visualizations. The interconnectedness is built based on the interactions between the various wearable IoT devices regarding information sharing. This can facilitate us to determine who contacted who. In each visualization, *SENS xxxxx*, *iPhone xxxxx*, and *Android xxxxx* represents a sensor with serial number, iPhone with serial number, and Android with serial number respectively. Also, there are some instances where we have some devices as *Unknown* which means these devices did not give their UUIDs or failed to subscribe with a unique identifier.

In Figure 7, the visual analytics shows the data origin and interconnectedness being represented in an RGraph composition with another RGraph (for node rendering).

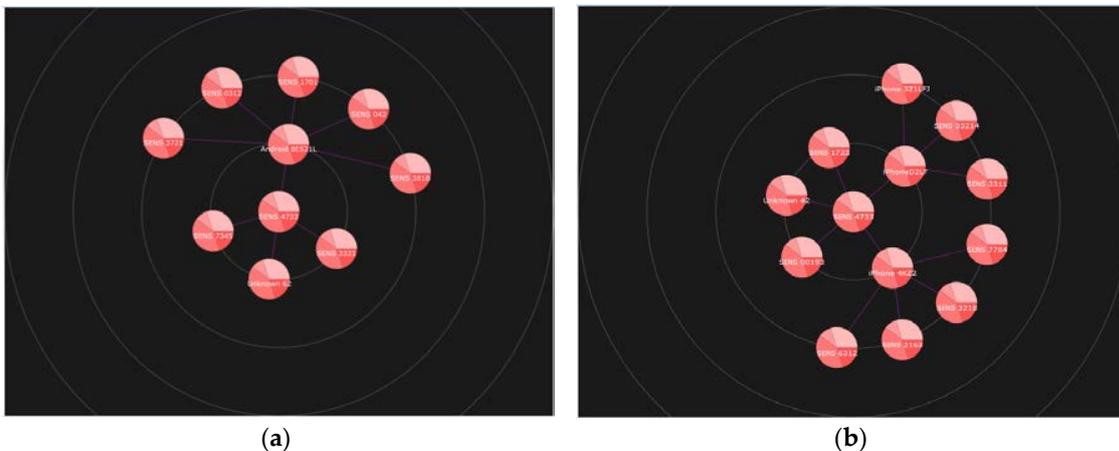


Figure 7. The visual composition of data trace routes showing which devices shared data according to the provenance record in (a,b).

For brevity, we shall explain the visualization on the left and the same logical flow applies to the visualization on the right. Showing the data shared to the n th degree, the n -value for this visualization is three (3). The parent node (in the inner circle), also called the root, is the sensor device labelled SENS 4733.

This means from the provenance record, a particular IoT data is traced from its origin, which in this case is the root, to the device that last acknowledged receipt of the information when the root broadcasted it. Clearly, the IoT data from the root device is received by four other devices which are SENS 7345, SENS 3321, Android 8E521L, and an unknown device whose serial number is recorded only as 62. However, the android 8E521L device shared the same information with five other devices which include SENS 3721, SENS 0312, SENS 1701, SENS 042, and SENS 381B.

The visual analytics provided data trace routes that transparently aids in identifying which devices are accessing the data in the wearable IoT. In this case, the identification of privacy breaches is made easier since users can easily identify suspicious nodes. A typical example is the “Unknown 62” device that in this case had accessed the data.

5. Conclusions

With the Internet-of-Things (IoT) field solidifying its place in our individual, corporate, and societal lives, there is a need to explore privacy questions. In wearable IoT for example, systems which are designed following the broadcast architecture for ubiquitous streaming are at risk. This is because these devices are always discoverable and susceptible to all forms of attacks.

Hence, this paper proposed the broadcast-subscriber IoT model where users' personal data are only shared with intended nodes such as healthcare facilities or devices authorized by the user. This means only devices with prior subscription approval will be able to easily consume the data. This is achieved by proposing two major approaches.

A consideration is given to the meta-data level encryption techniques where the unique identifiable components of the communicating devices (e.g., serial number, MAC address) are encrypted and the keys are only known to the devices within the subscription pool. This hardware and software level encryption technique means privacy breaches are minimal as shown in the various experiments. Several encryption-decryption algorithms are tested including the AES, DES, 3DES, MD5, and SHA.

Also, the provenance technique is proposed in the broadcast-subscriber IoT architecture to ensure transparency and full digital audit trail. The introduction of provenance facilitates the audit of the IoT data from source origin authentication to the current state. The interconnectedness of the devices is also investigated to determine who got what data within the wearable IoT architecture. Several evaluations are conducted which prove that the proposed system is less resource intensive, has a high propagation rate of the IoT data, highly secured through fault injection analysis, and visual analytics through the evaluation of the provenance record. Overall, the work accomplished:

- Provenance is proposed to generate data trace routes in the wearable IoT economy order to ensure digital audit trail for transparency. The work adopted the broadcast-subscriber IoT architecture to ensure privacy of users' data such as vitals especially in wearable IoT; a key concern in the generic broadcast IoT architecture.
- Both the hardware and software level data encoding methodologies are proposed based on device meta-data encryption.

The work will be made open source and available on GitHub. The extension of this work is the full integration into the health information system for real-time monitoring of vital stats by care providers. This stage will not be concerned much about privacy issues, since the current work has achieved that satisfactorily according to our industrial research partners.

Acknowledgments: The authors wish to thank Joseph Pry and Emmanuel Kaku for their contributions towards the initial development of the research. This work was supported in part by a grant from the Pennsylvania State University—Beaver, USA.

Author Contributions: Richard K. Lomotey: Lomotey developed the research concept and was responsible for the design and development of the project. Kenneth Sofranko: Kenneth works on the study of the gaps in the

research domain by identifying the key research questions that needed to be addressed through literature reviews. Rita Orji: Orji was instrumental in the research design methodology.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sicari, S.; Cappiello, C.; De Pellegrini, F.; Miorandi, D.; Coen-Porisini, A. A security-and quality-aware system architecture for Internet of Things. *Inf. Syst. Front.* **2016**, *18*, 665–677. [CrossRef]
2. Wang, P.; Valerdi, R.; Zhou, S.; Li, L. Introduction: Advances in IoT research and applications. *Inf. Syst. Front.* **2015**, *17*, 239–241. [CrossRef]
3. Li, N.; Sun, M.; Bi, Z.; Su, Z.; Wang, C. A new methodology to support group decision-making for IoT-based emergency response systems. *Inf. Syst. Front.* **2014**, *16*, 953–977. [CrossRef]
4. Zhang, Q.; Yang, L.T.; Chen, Z.; Li, P. High-order possibilistic c-means algorithms based on tensor decompositions for big data in IoT. *Inf. Fusion* **2018**, *39*, 72–80. [CrossRef]
5. Lomotey, R.K.; Nilson, J.; Mulder, K.; Wittmeier, K.; Schachter, C.; Deters, R. Mobile Medical Data Synchronization on Cloud-Powered Middleware Platform. *IEEE Trans. Serv. Comput.* **2016**, *9*, 757–770. [CrossRef]
6. Pencheva, E.; Atanasov, I. Engineering of web services for internet of things applications. *Inf. Syst. Front.* **2016**, *18*, 277–292. [CrossRef]
7. Atzori, L.; Iera, A.; Morabito, G. The Internet of Things: A survey. *Comput. Netw.* **2010**, *54*, 2787–2805. [CrossRef]
8. March, S.T.; Scudder, G.D. Predictive maintenance: Strategic use of IT in manufacturing organizations. *Inf. Syst. Front.* **2017**. [CrossRef]
9. Whitmore, A.; Agarwal, A.; Da Xu, L. The Internet of Things—A survey of topics and trends. *Inf. Syst. Front.* **2015**, *17*, 261–274. [CrossRef]
10. Lomotey, R.K.; Deters, R. Middleware for mobile medical data management with minimal latency. *Inf. Syst. Front.* **2016**, 1–16. [CrossRef]
11. Namboodiri, M. M2M and IoT—Security and Privacy Thoughts for 2015. Available online: <https://www.linkedin.com/pulse/m2m-iot-security-privacy-thoughts-2015-manu-namboodiri> (accessed on 1 August 2016).
12. Lomotey, R.K.; Sriramoju, S.; Deters, R. Middleware platform for the synchronisation of mobile medical data. *Int. J. Bus. Process Integr. Manag.* **2017**, *8*, 136–144. [CrossRef]
13. Pignotti, E.; Edwards, P. Trusted tiny things: Making the internet of things more transparent to users. In Proceedings of the International Workshop on Adaptive Security, Zurich, Switzerland, 8–12 September 2013.
14. Ziegeldorf, J.H.; Morchon, O.G.; Wehrle, K. Privacy in the Internet of Things: Threats and challenges. *Secur. Commun. Netw.* **2014**, *7*, 2728–2742. [CrossRef]
15. Bertino, E. Data Security and Privacy in the IoT. *EDBT* **2016**, *2016*, 1–3.
16. Shebaro, B.; Oluwatimi, O.; Midi, D.; Bertino, E. Identidroid: Android can finally wear its anonymous suit. *Trans. Data Priv.* **2014**, *7*, 27–50.
17. Singh, J.; Pasquier, T.; Bacon, J.; Ko, H.; Eysers, D. Twenty security considerations for cloud-supported Internet of Things. *IEEE Internet Things J.* **2016**, *3*, 269–284. [CrossRef]
18. Devi, K.N.; Muthuselvi, R. Secret Sharing of IoT Healthcare Data Using cryptographic algorithm. *Int. J. Eng. Res.* **2016**, *5*.
19. Lomotey, R.K.; Pry, J.; Sriramoju, S. Wearable IoT data stream traceability in a distributed health information system. *Pervasive Mob. Comput.* **2017**, *40*, 692–707. [CrossRef]
20. SensorTag. Available online: <http://www.ti.com/tool/cc2650stk#0> (accessed on 20 April 2018).
21. Xamarin. Available online: <https://www.xamarin.com/> (accessed on 20 April 2018).
22. CouchDB. Available online: <http://www.couchdb.com> (accessed on 20 April 2018).
23. Kura Open Source for IoT. Available online: <http://www.eclipse.org/kura/> (accessed on 20 April 2018).
24. Alharbi, K.; Lin, X. PDP: A Privacy-Preserving Data Provenance Scheme. In Proceedings of the 2012 32nd International Conference on Distributed Computing Systems Workshops, Macau, China, 18–21 June 2012; pp. 500–505. [CrossRef]

