Secure Lightweight Context-driven Data Logging for Bodyworn Sensing Devices

Muhammad Siddiqi†, Syed Taha Ali*, Vijay Sivaraman†
†*School of Electrical Engineering and Telecommunications*
†*University of New South Wales, Sydney, Australia*
**School of Electrical Engineering and Computer Science, NUST, Pakistan*
m.siddiqi@student.unsw.edu.au, taha.ali@seecs.edu.pk, vijay@unsw.edu.au

***Abstract*—Rapid advancement in wearable technology has unlocked a tremendous potential of its applications in the medical domain. Among the challenges in making the technology more useful for medical purposes is the lack of confidence in the data thus generated and communicated. Incentives have led to attacks on such systems. We propose a novel lightweight scheme to securely log the data from bodyworn sensing devices by utilizing neighboring devices as witnesses who store the fingerprints of data in Bloom filters to be later used for forensics. Medical data from each sensor is stored at various locations of the system in chronological epoch-level blocks chained together, similar to the blockchain. Besides secure logging, the scheme offers to secure other contextual information such as localization and timestamping. We prove the effectiveness of the scheme through experimental results. We define performance parameters of our scheme and quantify their cost benefit trade-offs through simulation.**

1. Introduction

Wearable sensing devices are fast becoming mainstream thanks to the success of smartwatches and fitness bracelets. Healthcare providers and major technology companies are also developing wearable health monitoring platforms. Examples include Google's partnership with Novartis to prototype a "smart contact lens" for wireless glucose monitoring and its healthcare platform Google Fit, Apple's partnership with Mayo Clinic and its HealthKit platform for the Apple Watch, and Samsung's healthcare platform called SAMI for its Galaxy series of smartphones. A recent PWC report finds that 20% of Americans already own a wearable device [1], the adoption rate for wearables, paralleling that of tablets, is expected to rise sharply over the next few years. IDC research company forecasts that worldwide sales of wearable devices will reach 213.6 million units by 2020 up from 79 million in 2015 and estimated 102 million in 2016 [2].

These devices have the potential to be more than just isolated sensors. With advances in data analytics and cloud computing, readings from individual sensors can be tracked over long periods, shared, and combined with other data to identify important trends and make informed diagnosis. This *contextualization* of data is expected to open new vistas for medical research and healthcare.

Health insurers are actively developing strategies to integrate bodyworn sensing devices into their policies. Firms such as John Hancock Insurance [3], United HealthCare Group [4] and MLC [5] now offer their customers free wearable sensing devices together with discounts on premiums and other financial incentives to keep active and meet wellness goals. However, for these devices to fully integrate into the existing medical infrastructure, patients, doctors, insurers, and other stakeholders must have strong confidence in data recorded by these platforms. A number of things can go wrong. The data might be corrupted. The wearable device may develop a fault. Hackers might attack the system and alter the data. The user herself may tamper with her data to claim benefits.

These concerns are not far-fetched: Researchers have also demonstrated various attacks on these devices including the ability to backfill medical data [6]. J&J recently issued a warning informing customers that one of their insulin pumps had a security vulnerability allowing a hacker to potentially administer a fatal insulin dose to users [7]. In 2016, an Australian insurer, CommInsure, colluded with doctors to alter and conceal patient medical records [8].

Research in the field has mostly focused providing individual security properties, such as data confidentiality, data integrity, and authentication. In this paper we present a data logging solution which secures the contextual relationships between sensor data streams in a lightweight manner. We leverage the fact that the density of smart devices in homes and buildings is increasing. Indeed, according to Gartner, a typical family home could contain more than 500 smart devices by 2022 [9]. In our solution smart devices in the same broadcast domain act as *witnesses* for neighboring sensors by logging all data transmissions (or *conversations*) that they overhear. However, instead of recording entire conversations, the sensors use Bloom filters to maintain fingerprints of conversations instead, thereby dramatically reducing the associated overheads. These fingerprints may later be examined for forensics purposes.

To further clarify the concept of witnesses, we consider an application scenario first presented by Prasad *et al.* [10]: a health worker, Devi, visits pregnant women in a village in India to perform medical checkups. She uses a

. This research is funded by Australian Research Council's Discovery Grant DP150100564.

mobile sensor kit consisting of blood pressure and heart rate monitors, weight scale, fetal monitor, spiral monitor, and smoke sensor. This sensor data is later uploaded into an electronic health record system at the village health clinic where it can be examined by doctors. There is a strong requirement here for a mechanism which preserves the contextual relationships between the data originating from the different sensors. For example, the system may inform the doctor that one patient is experiencing a decrease in lung capacity. The doctor may wonder if the spirometer is functioning correctly. However, the smoke sensor may show a strong concentration of nicotine in the patient's home which may be the actual cause. The system should also ensure that the sensor readings belong to the patient in question and they have not been tampered with in transit. This application scenario is equally applicable to hospitals.

In our solution, the gateway device will maintain a record of data it has received from all sensor devices during set time periods (or *epochs*). The smart devices in turn will maintain fingerprints of all data transmissions that they overhear.

In this paper, we make the following specific contributions:

1) We present a scheme that logs wearable healthcare sensor data in a secure and lightweight manner.
2) We present experimental results that confirm the effectiveness of the scheme.
3) We define performance parameters of the scheme and quantify their cost benefit trade-offs through simulation results.

The rest of the paper is organized as follows. §2 presents the background of this work. §3 describes our secure logging solution with the help of witnesses. §4 discusses the performance analysis of the scheme and we conclude the paper in §5.

2. Background

2.1. Prior Work

As we noted earlier, prior work for bodyworn sensing devices has mostly focused on primary security properties such as data confidentiality, data integrity, and authentication. Here we briefly highlight research in domains such as timestamping, authentication, and data provenance which has similarities to our solution:

Masdari *et al.* [11] provide a comprehensive survey of **authentication mechanisms** for bodyworn sensing devices. Ali *et al.* [12] in particular propose a method to amortize digital signatures to ensure integrity and non-repudiation for data recorded by bodyworn devices.

Digital timestamping originated in the work of Haber and Stornetta [13] and Pinto and Freitas [14]. Sundararaman *et al.* survey the techniques to improve the relative timestamps in wireless sensor networks [15], and, in the context of wearable healthcare devices, Siddiqi *et al.* [6] devise a protocol to authenticate timestamps on data packets.

Data provenance, i.e. the context in which data has been created and its evolution within a system, is another domain which bears resemblance to our solution. Prasad *et al.* [10] argue persuasively for a mechanism to preserve the contextual relationships for data originating from multiple sensor devices with a view to improving medical diagnoses. Ali *et al.* [16] propose a solution that binds data with a wireless link fingerprint between two communicating devices which may be used for forensics at a later time. Wang *et al.* [17] interpret provenance as the information regarding the path data takes from source to destination in a sensor network and secure this using Bloom filters.

On a related note, some researchers have proposed generalized **architectures** for communication and body area networks [18] [19] to enable certain properties but they do not focus on security or contextual relationships between sensor data as we do.

Recently, the blockchain has emerged as a popular **secure logging** solution. Developed by Satoshi Nakamoto in 2008 and used in Bitcoin, the blockchain maintains a secure distributed ledger of all Bitcoin transactions. Our solution loosely borrows from the blockchain architecture by chaining together blocks of readings from successive epochs to prevent retroactive tampering with data.

To the best of our knowledge, we are the first to propose a scheme for secure logging and forensics of medical data for bodyworn sensing devices. In our application, secure logging may be considered an umbrella term comprising a variety of important security properties, including information about the origins and integrity of the data, localization, timestamping, and chronological ordering of the individual data items. The use of Bloom filters on sensor devices ensures that the scheme is lightweight suited to run on resource-constrained devices. Furthermore, whereas our scheme does not explicitly provide data confidentiality, encryption mechanisms may easily be deployed independently of it to ensure privacy.

2.2. Bloom Filters

A Bloom filter is a space-efficient probabilistic data structure [20]. It is a compact way to store data where the requirement is to enquire membership and not to retrieve data. A query may provide a false positive with a probability, however, it cannot result in a false negative. A Bloom filter is indeed a bit array of a pre-defined size with all bits initialized to '0'. A data element that needs to be stored is hashed using one or more hash functions whose outputs are random and uniformly distributed over the indices of the bit array. Bits at the locations thus addressed are set to '1'. Same hash functions are used for membership query and the corresponding bits are checked. For a membership query to be "probable", all such bits must be set, however, a single '0' bit results in negative.

A Bloom filter is shown in Fig. 1 where m is the filter size in bits, n and k are the number of elements and hash functions respectively. Elements d_1, d_2, and d_3 are inserted in the filter using three hash functions h_1, h_2, and h_3. Here

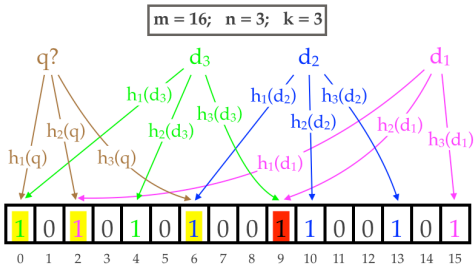

Figure 1. Bloom filter by example

bit at index '9' (red) is set twice as a result of collision. Note that another element q gives a false positive.

The probability of a false positive f is given below [21].

$$f \approx \left(1 - e^{-kn/m}\right)^k \quad (1)$$

Optimum value of k is given by: $k_0 \approx (m/n)ln2$

The filter size m for a given n and the probability of false positive f is given by:

$$m \approx -nln(f)/(ln2)^2 \quad (2)$$

A comprehensive survey on Bloom filters is found in [22].

3. Our Logging Solution

3.1. The Architecture

The architecture of our solution is depicted in Fig. 2. In our case, the gateway maintains a detailed log of all conversations that it conducts with sensor devices which it forwards in epoch-level blocks to a centralized server. These blocks are chained together, similar to the blockchain, i.e. each successive block contains a hash value of the previous block, and the chain is replicated in multiple locations to prevent retroactive data tampering. Sensor devices log all communications they overhear between other parties in the network and maintain a record which they later forward to the gateway. For this reason, we refer to these devices as *witnesses*.

However, we note a conflict: in case there is heavy traffic in the network, recording (and later communicating) all transmissions may pose unacceptably large memory and communication overheads for witnesses. Therefore we propose the witnesses use Bloom filters to log (or *fingerprint*) all transmissions they hear. Bloom filters considerably reduce the memory consumption and transmission overhead as well as ensure simple and accurate verification. However, in dense deployments of sensor devices, it is possible that multiple devices overhear communication from multiple sensors adding a lot to the cost. In order to further reduce logging overheads, it might be desirable that witnesses instead log the communication probabilistically.

It is assumed that the gateway and sensors use a time synchronization protocol at the start of every epoch. All sensors append a running counter value to every data transmission they overhear prior to inserting it into their Bloom filter, thereby enabling verification of the chronological ordering of the data in the gateway log, as well as a loose form of

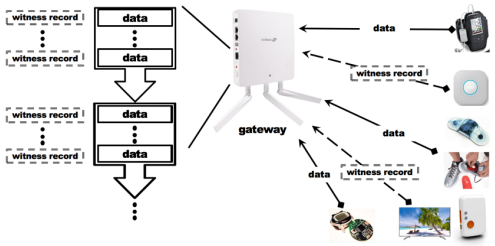

Figure 2. Architecture of witnesses logging scheme

localization or proximity. At the conclusion of each epoch, witnesses upload their Bloom filters to the centralized server. The fingerprint data is digitally signed by the witnesses, thereby enabling integrity and non-repudiation.

Forensics specialists may later use these fingerprints to certify the data in the detailed logs reported by the gateway. Given that there is packet loss in wireless networks, all sensors may not witness all data transmissions, however, as our experimental results indicate, there is a very high probability that a transmission is overheard by at least one witness.

3.2. The Experiment

We conduct an experiment with real wireless devices: a human subject wearing a MicaZ mote on his right arm (acting as a sensor device) walks for half an hour in a long office hall with cubicles (as depicted in Fig. 3 (Left)). One MicaZ mote acts as a gateway while three such motes act as witnesses, two are stationary while one is mobile similar to the subject. The sensor mote transmits at the rate of 1 pkt/sec at maximum transmission power which gateway logs. The witness motes in the range also log whatever they hear. Here, the number of packets are '$n = 1800$'. For the accuracy of 99% $(1 - f)$, we set the size of the Bloom filters to be m = 2.1 KB (using Eq. 2), and number of hash functions to be $k = 7$. We use python implementation of Murmur3 hash for Bloom filters.

The verification process uses the packets logged by the gateway mote and the three Bloom filters from the witness motes. The results are depicted in Fig. 3 (Right) where 97% packets are heard by at least one witness, which motivates our scheme and validates its effectiveness.

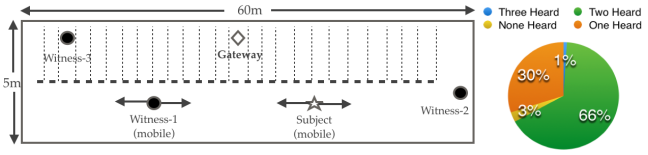

Figure 3. Layout of the experimental setup (Left). Results: Packets logged by the witnesses (Right)

3.3. Dimensioning for a Hospital Scenario

To better understand the scheme, let us consider the scenario of a hospital ward where the patients' vital signs are monitored by bodyworn sensing devices and the measurements are forwarded to a medical database through wireless

gateways. The monitored vital signs can be, but not limited to, temperature, blood pressure, blood glucose, heart rate, ECG values, respiration rate, and blood oxygen saturation.

Size of an acute care unit (ACU) of most of the hospitals vary from 10 to 100 beds while recommended ICU size is 8–12 beds [23]. It is reasonable to assume that a smart device such as a smoke detector, wireless access point, or any smart medical equipment in the hospital ward can be within the broadcast range of around 10 beds. Consider three vital signs (axillary temperature, heart rate, and respiration rate) are monitored from each patient including and transmitted to the gateway every 2 minutes (rate used by a medically approved device SensiumVitals System [24]). Under the given assumption, these smart devices hear approximately 30 packets every 2 minutes (a rate of 0.25 pkts/sec). In cases where there is more frequent reporting (e.g. ECG) or there are more sensors, however, this rate can reach or exceed 1 pkt/sec. Recall that an *epoch* is a time period after which witness devices forward their signed data (Bloom filters) to the gateway. Table in Fig. 4 lists the design choices (i.e. filter size and number of hash functions) for two different epoch lengths and four different bounds on the probability of false positive. Similarly, design choices can be worked out for other scenarios such as home, office, street, and transport.

Epoch Length	1 hour		12 hours		Optimal Hash Functions (k_o)
Probability of False Positive (f)	Size of Bloom Filter m (Bytes)		Size of Bloom Filter m (Bytes)		
	@ 0.25 pkt/s (n=900)	@ 1 pkt/s (n=3600)	@ 0.25 pkt/s (n=10,800)	@ 1 pkt/s (n=43,200)	
1%	1K	4.2K	12K	50K	7
5%	0.68K	2.7K	8K	33K	5
10%	0.5K	2.1K	6K	25K	4
20%	0.37K	1.5K	4.4K	18K	3

Figure 4. Bloom filter design choices for a hospital ward

It is apparent from the table in Fig. 4 that reducing the size of a Bloom filter for a given number of elements increases the probability of false positive. It is important to note that relaxing the probability of false positive from 1% to 5% gives the best trade-off in terms of memory and processing. In applications where storage and communication overhead are limited, Bloom filter size can be reduced if a greater probability of false positive can be tolerated. Moreover, decreasing the *epoch* length requires smaller Bloom filter (less storage) and hence lower communication overhead but increases the frequency of transmissions.

4. Performance Analysis

In this section, we define and discuss the performance parameters we use to evaluate our scheme. We also present and discuss the results from our simulation of the scheme.

4.1. Choice of Hash Function in Bloom filters

Hash functions play a fundamental role in Bloom filters and are chosen carefully depending upon the application. A hash function used in a Bloom filter must have its output uniformly distributed over the length of the Bloom filter as a fundamental requirement. Other parameters to consider while selecting a hash function are the security (cryptographic or non-cryptographic hash function) and computation cost. As the messages overheard by witnesses in a medical wearable scenario are most likely encrypted already, we may not need to use cryptographic hash functions. As wearable devices are resource-constrained, we need to reduce the computation cost of the hash function used. *Double hashing technique* reduces the cost of computation of k hash functions to only two independent hash functions as this technique allows us to generate more hash functions from only two independent hash functions. Popular choices of hash functions include MD5, Murmur3, CRC32, and SHA1. In our scheme, trust in an element's membership depends on how many witnesses overhear it. Use of different hash functions for different devices help reduce the probability of false positives further because, by doing so, an element causing a false positive in one Bloom filter is highly unlikely to produce a false positive in others.

4.2. Probabilistic Logging Parameter

Let W_i be the total number of witnesses available that hear the ith packet from a bodyworn sensing device. For probabilistic logging, the sensing device appends the count of number of devices it can hear to each packet. The witnesses then log the packets with a probability depending upon that number W_i after reading it from the packet. Let L be the event that a packet is logged by a witness, the probability of logging the ith packet is given by:

$$P(L_i) = \begin{cases} \frac{\beta}{W_i}, & \text{if } W_i > \beta \\ 1, & \text{otherwise} \end{cases} \tag{3}$$

Here β is the design parameter we refer to as **Probabilistic Logging Parameter** as it translates to the logging probability and determines how many witnesses ideally should log the data. Let w_i be the number of witnesses out of available W_i witnesses that successfully log the ith packet, where:

$$w_i = \begin{cases} \beta \text{ (on average)}, & \text{if } W_i > \beta \\ W_i, & \text{otherwise} \end{cases} \tag{4}$$

In summary, the average number of witnesses who log the data are:

$$\bar{w_i} \leq \beta \leq \bar{W_i} \tag{5}$$

If $P(L_i) = 1\ \forall i$, this we refer to as **All Witnesses Logging** compared to **Probabilistic Logging** in which $P(L_i) < 1$ for certain i in an epoch.

4.3. Cost of Logging

Let C_0 be the cost of logging a packet in a Bloom filter managed by a witness for this purpose. Cost to log the ith packet is given by:

$$C_i = C_0 w_i \tag{6}$$

Let T be the length of an epoch *(in seconds)* and $\bar{R}$ be the average transmission rate from the sensing device *(in pkts/sec)*, then total packets transmitted by the device in an

epoch are $T\bar{R}$. Total cost of logged data in an epoch is given by:

$$C = \sum_{i=1}^{T\bar{R}} C_i = C_0 \sum_{i=1}^{T\bar{R}} w_i \tag{7}$$

Average cost of logging data in an epoch becomes:

$$\bar{C} = C_0 \bar{w_i} \sum_{i=1}^{T\bar{R}} = C_0 \bar{w_i} T\bar{R} \tag{8}$$

This gives us an upper bound on average cost.

$$\bar{C} \leq C_0 \beta T\bar{R} \tag{9}$$

The cost of the Probabilistic Logging at its worst is equal to the cost of All Witnesses Logging, however, for most practical cases, it will be considerably less.

4.4. Accuracy of Verification

Let f_n be the probability of false positive in the Bloom filter managed by an nth witness, the accuracy of the membership query for the ith packet is defined as below:

$$A_i = 1 - \prod_{n=0}^{w_i} f_n \tag{10}$$

Here, we define $f_0 = 1$ to get $A_i = 0$ when $w_i = 0$, that is the accuracy of the membership query when there are no witnesses to log the packet.

Accuracy of the data logged in an epoch can be calculated by averaging out the accuracy values for individual packets over all the packets in an epoch:

$$\bar{A} = \frac{1}{T\bar{R}} \sum_{i=1}^{T\bar{R}} A_i = 1 - \frac{1}{T\bar{R}} \sum_{i=1}^{T\bar{R}} \prod_{n=0}^{w_i} f_n \tag{11}$$

As it may seem that the accuracy of verification in Probabilisitic Logging is lower than the one in All Witnesses Logging (equal at best), however, it is important to note that the probability of false positive f_n for the former case is usually far less than that of the later case (equal at worst). This is due to the fact that for the given size of a Bloom filter and number of hash functions, number of packets logged determine the probability of a false positive. Since in Probabilistic Logging, witnesses do not log all the packets, it decreases the probability of false positive. Hence the accuracy does not reduce drastically.

4.5. Trust in the Validity of Data

Our confidence in a packet logged by the gateway increases if witnesses log it as well. Given the fact that there is a certain probability of false positive in a Bloom filter, trust has to increase with the number of witnesses. Trust in the validity of ith packet is defined as below:

$$\tau_i = 1 - e^{\frac{-w_i}{\alpha}} \tag{12}$$

Here, α is a design parameter we refer to as **Trust Defining Parmater**, which is used to define the value of trust τ with respect to the number of witnesses. For $\alpha > 1$, the trust values drop for a given number of witnesses while for $\alpha < 1$, they increase. The choice of α depends upon the requirement whether there is a need for more witnesses or one or two suffice. If the probability of false positive is greater, it is a good idea to set $\alpha > 1$ as more witnesses are required for a better trust. (refer to Fig. 5)

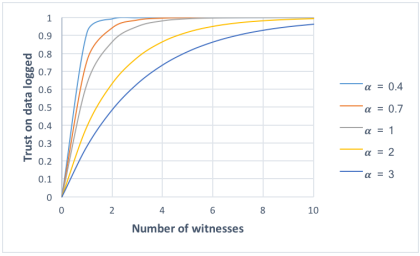

Figure 5. Trust in logged data vs number of witnesses at varying α

Trust in the data logged in an epoch can be calculated by averaging out the trust values for the individual packets over all the packets in the epoch:

$$\bar{\tau} = \frac{1}{T\bar{R}} \sum_{i=1}^{T\bar{R}} \tau_i = 1 - \frac{1}{T\bar{R}} \sum_{i=1}^{T\bar{R}} e^{\frac{-w_i}{\alpha}} \tag{13}$$

Trust value in Probabilistic Logging is lower than the one in All Witness Logging (equal at best). However, the true comparison of trust between the two cases depends upon the selection of Trust Defining Parameter α. Trust should always be read alongside accuracy because it compensates for a lower value of trust. For example, for a lower trust value that indicates that one witness has logged a packet, if the accuracy is very high (corresponding to very low probability of false positive), it calls for a positive verification.

4.6. Simulation Results

We simulate our scheme using the Python language where a sensing device transmits at the rate of 1 pkt/sec. Different number of witnesses (ten at maximum) at random log the data at varying probabilities depending upon the number of witnesses available for each packet (value of W_i in each packet). Length of the epoch is set to be 1 hour, so total number of packets in an epoch are 3600. Size of each Bloom filter managed by witnesses is set to be 22,447 bits $\approx$ 2.75 KB. The Bloom filter size is set targeting the accuracy $(1 - f)$ of 99.5% (Eq. 2), if a witness were to log all the packets in an epoch. Five Murmur3 hash functions are used to log a packet in the Bloom filter.

We log the sensing data of an epoch for different values of Probabilistic Logging Parameter $(0 \leq \beta \leq 10)$, which translates to the probability of logging (refer to Eq. 3). Graph in Fig. 6 shows the average cost of logging data in an epoch for different values of Probabilistic Logging Parameter (β). Graph in Fig.7 shows the average accuracy of verification of data in an epoch (refer to Eq. 11) while graph in Fig. 8 provides the average trust in data (at $\alpha = 1$) in an epoch (refer to Eq. 13) for different values of Probabilistic Logging Parameter (β). From these graphs it is apparent that

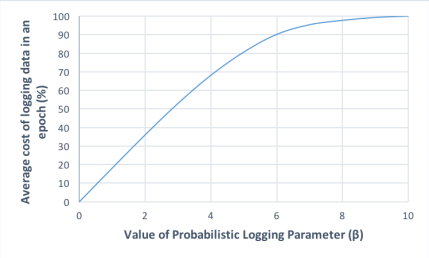

Figure 6. Average cost of logging data in an epoch at varying values of the Probabilistic Logging Parameter (β)

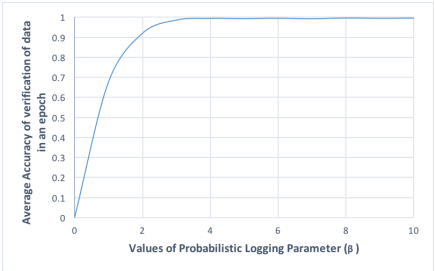

Figure 7. Average accuracy of verification of the logged data of an epoch with varying values of the Probabilistic Logging Parameter (β)

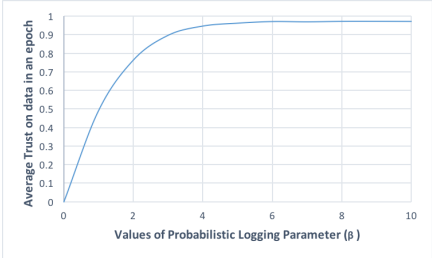

Figure 8. Average trust in the logged data of an epoch with varying values of the Probabilistic Logging Parameter (β); (at $\alpha = 1$)

the Probabilistic Logging offers a much more cost effective solution than that of All Witnesses Logging scheme. In fact, setting Probabilistic Logging Parameter (β) value to 3 gives an accuracy of 98.5% and a trust of 90% with a huge 47% savings in the cost and offers a desirable alternative.

5. Conclusion

Wearable technology has the potential to bring about a major change in how the healthcare system works today. Stakeholders including healthcare providers, insurers, and patients are reluctant to shift to this emerging ecosystem because of the successful attacks on such systems by hackers surfacing severe underlying vulnerabilities. Assurances on the authenticity and integrity of data from these devices are the key requirements. To meet these requirements, we presented a scheme that leverages the presence of neighboring smart devices we referred to as witnesses and provides a lightweight secure data logging solution. The solution amortizes cost of logging using Bloom filters and stores chronologically ordered data of forensic significance in a blockchain fashion. We motivated our idea with promising results from our experiment with real wireless devices. We defined performance parameters of our protocol and illustrated their cost benefit trade-offs through simulation results. Our scheme is the first step towards secure logging and forensics of medical data for bodyworn sensing devices.

References

[1] PwC, "The Wearable Future," https://goo.gl/7kDrxh, 2015, [Online; accessed 13-June-2016].

[2] Wearable, "Welcome to the 100 million club: Wearable sales soar in 2016," https://goo.gl/ZG3dIZ, [Online; accessed 04-Feb-2017].

[3] J. H. Insurance, "John Hancock Introduces a Whole New Approach to Life Insurance in the U.S. That Rewards Customers for Healthy Living," https://goo.gl/DV7a3M, [Online; accessed 13-June-2016].

[4] UnitedHealthcare, "UnitedHealthcare and Qualcomm Collaborate to Launch New Wellness Program That Links Financial Incentives with the Use of Wearable Devices," https://goo.gl/R0DUiY, March 1, 2016, [Online; accessed 13-June-2016].

[5] I. Big Cloud Analytics, "National Australia Bank's "MLC On Track" Program Leverages Big Cloud Analytics' Predictive Analytics from Wearable Devices and Internet of Things Data," https://goo.gl/MlxhdK, March 1, 2016, [Online; accessed 13-June-2016].

[6] M. Siddiqi, V. Sivaraman, and S. Jha, "Timestamp Integrity in Wearable Healthcare Devices," in *IEEE ANTS, Banglore, India*, Nov. 2016.

[7] J. Finkle, "J&J warns diabetic patients: Insulin pump vulnerable to hacking," https://goo.gl/oznLsM, [Online; accessed 13-June-2016].

[8] R. Fogarty, "CommInsure: Who's who in the Commonwealth Bank's life insurance scandal?" https://goo.gl/edw79r, Mar. 2016, [Online; accessed 13-June-2016].

[9] icontrol Networks, "2015 State of the Smart Home Report," https://goo.gl/7xU5RW, 2015, [Online; accessed 13-June-2016].

[10] A. Prasad, R. A. Peterson, S. Mare, J. Sorber, K. Paul, and D. Kotz, "A provenance framework for mhealth." in *COMSNETS*. IEEE, 2013.

[11] M. Masdari and S. Ahmadzadeh, "Comprehensive analysis of the authentication methods in wireless body area networks," *Security and Communication Networks*, 2016.

[12] S. T. Ali, V. Sivaraman, and D. Ostry, "Authentication of lossy data in body-sensor networks for cloud-based healthcare monitoring," *Future Generation Computer Systems*, vol. 35, pp. 80–90, 2014.

[13] S. Haber and W. S. Stornetta, "How to time-stamp a digital document," *Journal of Cryptology*, vol. 3, pp. 99–111, 1991.

[14] F. Pinto and V. Freitas, "Digital time-stamping to support non repudiation in electronic communications," in *the SECURICOM 96*, 1996.

[15] B. Sundararaman, U. Buy, and A. D. Kshemkalyani, "Clock synchronization for wireless sensor networks: a survey," *Ad hoc networks*, vol. 3, no. 3, pp. 281–323, 2005.

[16] S. T. Ali, V. Sivaraman, D. Ostry, G. Tsudik, and S. Jha, "Securing first-hop data provenance for bodyworn devices using wireless link fingerprints," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 12, pp. 2193–2204, 2014.

[17] C. Wang, W. Zheng, and E. Bertino, "Provenance for wireless sensor networks: A survey," *Data Science and Engineering*, vol. 1, no. 3, pp. 189–200, 2016.

[18] M. Chen, S. Gonzalez, A. Vasilakos, H. Cao, and V. C. M. Leung, "Body area networks: A survey," *Mobile Networks and Applications*, vol. 16, no. 2, pp. 171–193, 2011.

[19] S. Ullah *et al.*, "A comprehensive survey of wireless body area networks," *Journal of Medical Systems*, vol. 36, no. 3, 2012.

[20] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, pp. 422–426, 1970.

[21] A. Broder and M. Mitzenmacher, "Network Applications of Bloom Filters: A Survey," *Internet Mathematics*, vol. 1, no. 4, 2003.

[22] S. Tarkoma, C. E. Rothenberg, and E. Lagerspetz, "Theory and practice of bloom filters for distributed systems." *IEEE Communications Surveys and Tutorials*, vol. 14, no. 1, pp. 131–155, 2012.

[23] Bertolini *et al.*, "The relationship between labour cost per patient and the size of intensive care units: a multicentre prospective study," *Intensive Care Medicine*, vol. 29, no. 12, pp. 2307–2311, 2003.

[24] Sensium, "SensiumVitals System: Wireless monitoring of vital signs," https://goo.gl/nHvUs0, [Online; accessed 06-Jan-2017].