This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2927674, IEEE Transactions on Dependable and Secure Computing

Secure Opportunistic Contextual Logging for Wearable Healthcare 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@unsw.edu.au, taha.ali@seecs.edu.pk, vijay@unsw.edu.au

Abstract—Wearable technology is increasingly being used for medical applications such as continuous monitoring of chronically ill patients in homes and hospitals. The various stakeholders (patients, doctors, insurers) have an interest in ensuring not only that the data is untampered, but also that the context is verifiable (e.g. correct time and location can be associated with the data collected). Prior works have studied these aspects in isolation, typically using cryptographic techniques. In this paper, we develop a new solution that leverages the density of wireless devices in the vicinity of the transaction to create witness records ensuring data is tamper-protected and bound to its time and location context. Our first contribution is to develop a secure logging architecture that compacts witness records using Bloom filters and hash-chains them to bind them to the data, allowing fast and reliable forensic verification. Our second contribution is to identify the various configuration parameters influencing the performance of our scheme in terms of storage, processing, and transmission efficiency, and to quantify their effect on verification accuracy. Our third contribution implements and demonstrates the feasibility of our scheme, and quantifies its efficacy via simulation using real trace data from a multi-storey building representing a hospital environment.

1 Introduction

Wearable sensing devices are fast becoming mainstream as evidenced by the success of smartwatches and fitness bracelets. Healthcare providers and major technology companies are also developing wearable health monitoring platforms. Examples include Google's healthcare platform, Google Fit [1], Google's collaboration with Novartis to prototype a "smart contact lens" for wireless glucose monitoring [2], Apple's partnership with Mayo Clinic and its HealthKit platform for the Apple Watch [3], and Samsung's SAMI healthcare platform for its Galaxy series of smartphones [4]. A recent PWC report finds that 20% of Americans already own a wearable device [5], and it is expected that the adoption rate for wearables will parallel that of computer tablets and rise sharply over the next few years. IDC research company [6] forecasts that worldwide sales of wearable devices will reach 213.6 million units by 2020 up from 79 million in 2015. Moreover, in 2016, total shipments of wearable devices were estimated to be 106 million with 38.6 million wearable therapeutic medical devices [7].

The development and adoption of wearable devices has to-date been largely driven by recreational use, such as for sports and fitness training. However, medical providers and insurers are increasingly looking to use these devices as a cost-effective and practical solution to cope with the healthcare needs of a rapidly ageing population and to combat an epidemic of chronic conditions such as diabetes, obesity, and heart disease [8]. Wearable sensors allow doctors to remotely treat patients in distant and rural communities, a service of particular importance in developing countries. Since these devices can continuously collect data, that can then be analyzed using powerful cloud-based tools, patient condition and lifestyle can be tracked over long periods, and combined with other data from multiple sources to identify important trends and make informed diagnosis.

Health insurers are actively developing strategies to integrate wearable sensing devices into their policies. Firms such as John Hancock Insurance [9], United HealthCare Group [10] and MLC [11] now offer their customers free wearable sensing devices together with discounts on premiums and other financial incentives to stay 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 the data recorded by these platforms. A number of things can go wrong: the wearable device may develop a fault and its data might be corrupted; hackers may attack the system and alter the data; the user herself may tamper with her data to claim benefits.

Hackers have demonstrated they can break into wearable healthcare devices ranging from pacemakers [12], [13] to insulin pumps [14], [15] and disable them or make them malfunction. Data tampering is also known to be feasible and prevalent – our prior work has shown that patients can backfill readings taken using approved medical wearable devices [16], while doctors and insurers were found to be colluding to alter patient medical records [17]. This reinforces the need to secure not just the wearable device, but also the data and the context associated with the measure-

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

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2927674, IEEE Transactions on Dependable and Secure Computing

ments, even from internal stakeholders including patients, doctors, and insurers.

While cryptographic techniques like encryption and digital signatures can ensure confidentiality and authenticity of messages, they do not provide any assurances regarding the context of the data, such as whether the measurement was taken at the stated location at the stated time. A new approach is therefore needed that can ensure contextual correctness, in addition to message confidentiality/authenticity that existing cryptographic techniques provide. Recent approaches for securing contextual metadata associated with wearable devices have either focused on verifying the last-mile link [18] or on the network path [19], and do not address aspects such as time and location.

In this paper we present a data logging solution which secures the contextual relationships between streams from multiple devices in an intuitive and lightweight manner. We leverage the fact that the density of smart devices in homes and buildings is rapidly increasing[1]. In our solution, devices in the same broadcast domain act as *witnesses* for each other's communications by logging all data transmissions (or *conversations*) that they overhear. Needless to say, recording these entire conversations can be prohibitively expensive for wearable devices that have very limited compute power, device memory, and battery life [21]. The major challenge is to compact the information to improve efficiency, while still retaining important information that allows forensic verification of the transactions noted in the data logs.

Our solution provides numerous interesting security properties. As we detail in §2.1 and §3.3, our solution enables non-repudiation and protects from retroactive data tampering. This crowdsourcing approach to securing data logs also enables efficient timestamping and verification of device location or proximity. The chronological ordering of data is strictly maintained. Overall, this approach secures the contextual relationship between various data items. The use of Bloom filters also ensures our scheme has minimal processing overhead. We make the following specific contributions in this paper:

1) We present a secure yet lightweight solution that uses wireless smart devices in the vicinity of the wearable device as witnesses to the transaction, thereby providing opportunistic binding of the medical data to its context.
2) We evaluate the impact of system parameters e.g. epoch length and Bloom filter size on performance metrics such as false positives and cost of our solution. We further show how our system can be further optimized for varying density of deployment.
3) We implement our scheme on MicaZ motes to demonstrate feasibility, and undertake a simulation study of our scheme using real data collected from a multi-storey building representing a large hospital environment.

Our experimental results indicate that our scheme can verify up to 98.7% of data from bodyworn sensing devices with up to and beyond 99% confidence levels with moderate processing, memory, and transmission overheads on the part of witness devices. We further show that greater saving may be achieved by fine tuning the scheme parameters. For example, a 37% saving in processing cost and memory may be attained without reduction in confidence level at the expense of the verifiability of a small number of packets ($<$1%) in a sparse sensor deployment scenario.

The rest of the paper is organized as follows: §2 presents background material to motivate our solution. §3 describes our secure logging solution in depth together with a discussion of security properties. §4 illustrates the configuration options and performance metrics of our solution. We discuss the experiments and results in §5 and conclude in §6.

2 Background

2.1 Secure Logging

Maintaining data and event logs of network activity is recommended by NIST as a key component of incident response [22] and is now a commonplace practice in industry. Computer logs act as a valuable historical record of system activities and, in the event of an incident such as a data breach or network failure, enable forensics specialists to undertake post-event analysis and identify the precise cause of the failure. Data logs may even serve as vital evidence in a court of law.

In the context of healthcare, data logs may even assist medical practitioners in patient diagnosis. Here we consider an application scenario, first presented by Prasad *et al.* [23], describing an e-health deployment in a remote Indian village: a health worker, Devi, visits pregnant women in a village to perform medical checkups. She uses a 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. This application scenario is equally applicable to other healthcare deployments.

In this instance, 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. There is therefore a strong requirement here for a data logging solution which preserves the chronological and contextual relationships between the data originating from the different sensors and enables medical specialists to make an accurate diagnosis.

Furthermore, the logs themselves need to be secured in most instances. Hackers routinely employ a variety of'anti-forensics' techniques to cover their tracks, which include altering data logs to evade detection [24]. As we noted earlier, users themselves may tamper with the data originating from their devices. This scenario is not too remote: data from bodyworn exercise and fitness devices has begun to feature as key evidence in criminal investigations [25], even as researchers are demonstrating the ease with which these devices may be hacked [26].

1. According to Gartner, a typical family home could contain more than 500 smart devices by 2022 [20]

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2927674, IEEE Transactions on Dependable and Secure Computing

Our key insight is based on the fact that wireless devices are generally not deployed in isolation in hospitals or homes, and different devices on the same network can act as bona fide 'witnesses' that can corroborate each other's data communications. Secure logging, in our application, may therefore be considered an umbrella term comprising a variety of important security properties. These include information about the origins and integrity of sensor data and its chronological ordering. Proof of wireless association may also serve as a loose form of localization. Together, these properties secure the vital contextual relationships within the data logs which are essential to detailed diagnosis and forensics. Our use of Bloom filters ensures that the scheme is lightweight and suited for resource-constrained devices. Furthermore, whereas our scheme does not explicitly provide data confidentiality, lightweight symmetric key encryption mechanisms such as AES may be deployed independently to ensure privacy. We discuss these security properties in greater detail in §3.3.

2.2 Bloom Filters

A Bloom filter is a space-efficient probabilistic data structure [27] used for compact data storage where the primary requirement is membership enquiry and not data retrieval. A query may provide a false positive with a certain probability, however, it cannot result in a false negative. The filter is 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 array. Bits at the locations thus addressed are set to '1'. For membership query, the same hash functions are used on the data item and the corresponding bits are checked. For a membership query to be "probable", all such bits must be set. However, one or more '0' bits result in a negative outcome.

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

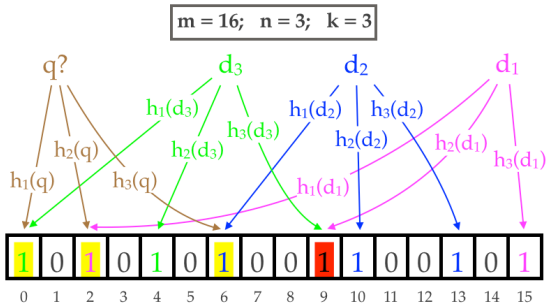

Fig. 1. Bloom filter by example

in the filter using three hash functions h_1, h_2, and h_3. Ideally it should result in nine distinct bits to be set, however, due to possible collisions, certain locations are set more than once. Here bit at index '9' (highlighted in red) is set twice as a result of collision. Later if these data elements d_1, d_2, and d_3 are queried by using same hash functions and checking the corresponding bits, their memberships can be verified. However, another data element q, when queried, shows positive for membership falsely as all three hash functions point to the bits separately set for different elements (highlighted in yellow).

The probability of a false positive f is given below [28].

$$f = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^k \tag{1}$$

By using the approximation from [28] in Eq. 1 leads to:

$$f \approx \left(1 - e^{-kn/m}\right)^k \tag{2}$$

To compute the optimal value of k, the probability of false positive $\left(1 - e^{-kn/m}\right)^k$ is minimized with respect to k. Taking its derivative with respect to k, equating to zero, and solving for k leads to the optimal value k_0. Optimum value of the number of hash functions k is given by:

$$k_0 \approx (m/n)ln2 \tag{3}$$

Here k_0 is rounded off to the closest integer. Using the optimal value of k from Eq. 3 in Eq. 2 and solving for m leads to the following expression.

$$m \approx -nln(f)/(ln2)^2 \tag{4}$$

Eq. 4 is used to determine the filter size m for an application for a given number of data elements n and the probability of false positive f. A comprehensive survey on Bloom filters is found in [29].

2.3 Prior Work

Here we briefly discuss prior work in the research literature on secure logging. Secure logging protocols may be broadly categorized into four categories, namely Syslog, Schneier-Kelsey, Ma and Tsudik, and Encrypted Search class. The interested reader is referred to a comprehensive survey [30].

The original **syslog** protocol was not designed considering security. Some security features are added in its later extensions, which include: syslog-ng (new generation) that supports encrypted transmission; syslog-sign that uses digital signatures to feature non-repudiation and data integrity; and "reliable syslog" aiming to provide successful delivery of syslog messages. Syslog class of protocols consider the collector honest, which does not align with our threat model. Adding encryption can provide data confidentiality against eavesdroppers but does not protect against malicious medical/insurance server and not even against malicious gateway in many scenarios where gateway is allowed to decrypt and show the readings to the patient. Digitally signing all messages poses unacceptably large processing overhead over the sensor device and is not suitable for our case.

Schneier and Kelsey [31] designed a protocol which secures stored data by use of time variant keys. These keys are created by hashing previous data contents and are updated every epoch such that the compromise of a key only compromises data for that particular epoch. This technique enables data authentication and evidence of data tampering by identifying broken links in the sequence of keys.

There have been several improvements to Schneier and Kelsey's scheme. For instance, Holt [32] modifies Schneier

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2927674, IEEE Transactions on Dependable and Secure Computing

and Kelsey's solution for a public key setting. Accorsi [33], addresses vulnerabilities to certain tampering attacks by deploying trusted hardware, and notably enhances the protocol for secure transmission of data logs by encrypting and signing each message and use of sequence numbers and timestamps. Whereas most of the protocols in this category aim to secure the chronological ordering of data items in addition to the data itself, the use of digital signatures and meta-data for individual data items is not suitable for small resource constrained devices due to the associated processing and transmission costs [21].

The next family is based on the work of **Ma and Tsudik** [34] which improves upon the storage and communication costs of prior logging schemes. Ma and Tsudik's protocol makes use of Forward Secure Sequential Aggregate (FssAgg) authentication techniques in which the signatures of a signer are sequentially combined to form a single aggregate signature. Verifying the aggregate signature verifies all the generated signatures while unsuccessful verification indicates that one or more signatures are not valid and the data has been tampered with. However, they assume the logger is honest so his signatures and the sequential aggregate signatures provide forward security, data integrity and protection against re-ordering of messages by any malicious party who tries to modify or re-order the log. This also does not align with our threat model, where the collector (medical/insurance server) cannot be trusted. As this scheme does not address the security during transmission, if it somehow be combined with a secure transmission protocol, the sensor will have to take the role of a logger, which would again pose unacceptably large processing overhead over sensor with frequent digital signatures.

The **Encrypted Search** category consists of schemes in which log data is encrypted. The challenge in this case is to retrieve encrypted data efficiently without decryption which risks exposing classified data and incurs high computational cost. Representative protocols in this class include the work of Waters *et al.* [35] who uses Identity Based Encryption (IBE) to search encrypted audit trails in which a designated trust party builds keyword search capabilities for investigators. On the other hand, Ohtaki [36] employs Bloom Filters to partially reveal audit trails together with search capability using AND and OR operators. Searches return probabilistic results indicating the likelihood of a match. This class of protocols does not secure data during transmission.

The majority of security solutions for wearable sensing devices focus on security properties such as bootstrapping secure communications, confidentiality, data authentication, and integrity of data [21]. However, some solutions provide a subset of the properties that we aim for in our solution. For instance, our work in Ali *et al.* [37] propose a method to ensure integrity and non-repudiation of data recorded by bodyworn devices by amortizing digital signature costs using Merkle trees. Likewise, our work in Siddiqi *et al.* [16] devise a protocol to authenticate timestamps on data packets. Certain solutions enable post-event forensics by efficiently securing data provenance, i.e. information regarding the origins and path that data takes in the network from source to destination [18] [38].

We are aware of only one secure logging solution for wearable healthcare devices in the literature: De La Piedra *et al.* [39] present a scheme in which a gateway node links messages from multiple wearable devices and constructs a 'threaded authentication tree' (an extension of the Merkle tree) by hash chaining relative timestamps of the messages. A central server then further links the authentication trees from individual gateway nodes to form a system-level authentication tree. However, readings from one gateway cannot be compared to those of other gateways because the timestamps are independently maintained and are not globally synchronized. The protocol claims to provide authentication, confidentiality, and data integrity. However, it does not provide protection where the linking authorities, namely gateways and central servers are malicious.

In contrast to the aforementioned schemes, our scheme is based on a new principle, that of attestation by multiple independent remote witnesses. This provides a check against malicious logging authorities, and also secures log contents during transmission and in storage. This not only secures the content but also the context of data originated from a wearable sensor device including time, location, and the environment. Other sensors in the vicinity taking part in this scheme can provide a detailed and verifiable snapshot of the environment that could help a doctor diagnose patient's condition with a better understanding of the surroundings. Furthermore, our scheme uses Bloom filters, thereby incurring significantly low computation and storage overheads, ideally suited for resource-constrained devices.

We first explored this idea of using neighboring devices as witnesses in a short paper [40]. In this paper, we undertake a more rigorous treatment, where we refine the security definitions, specify new evaluation metrics, undertake a detailed security analysis of our scheme, and perform a thorough experimental evaluation by modeling a hospital-like environment using real-world packet traces.

3 Our Logging Solution

In this section we present threat model and a detailed description of our secure logging solution, followed by a discussion of its security properties and certain practical concerns.

In our scenario, a gateway device, typically a WiFi access point or a smartphone, communicates with multiple wearable devices in a star topology. The gateway maintains a complete log of all its communications. Since these exchanges occur over the wireless link, other devices in the vicinity, including other wearable and non-wearable IoT devices such as thermostats, (called *witnesses*) are in a position to overhear them and maintain a local record of this communication. These witness records serve as a strong and independent check against malicious actors who may seek to alter or falsify previous records. A medical server and/or insurance server in the cloud records the data originating from users' wearable devices. The motive of our solution is to secure the sensor's communication from the adversaries, who could change its contents (physiological readings such as blood pressure, blood glucose levels) or the context such as time and location of data generation, while in transit (at the gateway) or at its destination (medical/insurance servers).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2927674, IEEE Transactions on Dependable and Secure Computing

3.1 Threat Model

Attacker #1 could be the owner of the sensor device who wants to inject false data to the medical/insurance server to enjoy certain benefits. For instance, a Canadian law firm used a client's Fitbit activity records in a personal injury compensation case to successfully prove that she had reduced activity levels even after four years of suffering injury [41]. Attacker's capabilities and limitations are summarized as follows:

- The attacker could compromise the gateway to either inject the false data to the cloud or generate the fake gateway log.
- The attacker could compromise the wearable sensor device retroactively to change its stored logs or use the sensor device to backfill medical data by compromising the gateway, which is usually relied for the timestamps. Such an attack is demonstrated in our previous work [16] where we backfilled data to the cloud from two devices: a smart blood pressure monitor "Fora Diamond Cuff BP" and a fitness tracker "Withings Pulse O_2. We have also shown that their tampered smartphone apps worked between the sensors and their servers well without raising any flags.
- The attacker could launch an attack at run-time (at the time of data generation) or attack retroactively.
- The attacker cannot compromise the witnesses or cannot compromise enough of them. The wearable and IoT devices in the vicinity might not be owned by the attacker. Moreover, the number of witness devices to compromise could be too high requiring a lot of effort. Furthermore, in case of probabilistic logging, the attacker cannot know which device will be witnessing which packet.

From attacker #1 capabilities, we conclude the following: wearable sensor device is a semi-trusted entity i.e. its records can be tampered retrospectively; gateway is malicious and cannot be trusted; witnesses are trusted entities and the only distrust in the verified data comes from their probability of false positive.

Attacker #2 could be healthcare and insurance providers, who have incentives to tamper with the data in order to e.g. conceal a bad diagnosis or reject an insurance claim respectively. They can do so retroactively at the storage phase once the data has been collected. This renders medical and insurance servers malicious and untrusted entities.

It is important to note that the scheme does not let the forensic investigator launch any attack even if he has any incentive to do so.

3.2 The Protocol

The architecture of our solution is depicted in Fig. 2. The protocol may be divided into the following key processes: maintenance of the gateway log, the witness record, and the verification process.

3.2.1 Gateway Log

The gateway maintains a complete log of all communications that it undertakes with wearable devices, which we refer to as the *gateway log*. After each *epoch* (a pre-configured time interval), Δt, the gateway digitally signs and forwards epoch-level data blocks to a medical server for processing and storage.

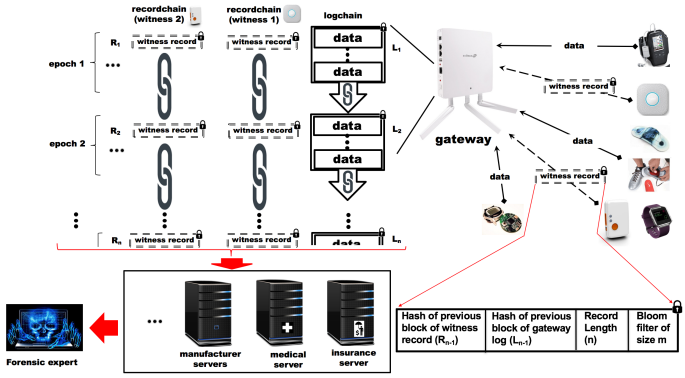

Fig. 2. Architecture of witnesses logging scheme

To protect against retroactive tampering of the data, we include two steps that are common in the secure logging literature: first, each successive data block of the log contains within it the hash value of the previous block in the temporal sequence, thereby ensuring a continuity similar to the Bitcoin blockchain [42]. Second, this gateway log is also periodically replicated at the servers of the health insurer and other stakeholders.

3.2.2 Witness Record

Since all communication between gateway and wearable healthcare devices is conducted over the wireless medium, other parties in the vicinity can overhear and record the exchanges. We refer to these devices as *witnesses* and the records they maintain as the *witness record*, which they later sign and forward to the medical and insurance servers. Witness records are maintained in synchronization with the epoch-level data blocks of the gateway log as shown in Fig. 2. It is important to note here that the witness record is agnostic to the nature of the traffic that is overheard. Witnesses may therefore include all other devices sharing the medium, including other wearable devices, smartphones, assorted wireless IoT devices, such as Internet-enabled thermostats, and even other gateways.

The challenge involved in this case however is that recording (and later communicating) all overheard messages may pose unacceptably large memory and communication overheads for witness devices, especially if they are resource-constrained wearable devices. Therefore we propose the witnesses use Bloom filters to store a *fingerprint* of messages they overhear.

Bloom filters considerably reduce the memory consumption and transmission overhead as well as ensure simple and highly accurate verification (as discussed in §2.2). In some cases, the overheads may be even further reduced, as for example in areas with a large number of witnesses, by enabling them to *probabilistically* witness communication they overhear (discussed in §4.3).

We note here another challenge: whereas Bloom filters enable verification of items within a set, they do not allow verification of the temporal ordering of the items. To ensure

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2927674, IEEE Transactions on Dependable and Secure Computing

the witness record also secures the chronological ordering of data in the gateway log, we recommend use of a network-wide time synchronization protocol. Various protocols have already been proposed in the literature for this purpose, such as Reference Broadcast Synchronization and Timing-Sync Protocol for Sensor Networks [43].

In our solution, time intervals are synchronized as per the gateway's epoch. At the start of every epoch, the gateway broadcasts a synchronization beacon which forces all listening devices to set a counter at their end[2]. The devices also respond to the beacon, thereby enabling the gateway to maintain a list of potential witnesses which is later forwarded to the medical server along with the data in every epoch. After this synchronization process, whenever a witness device overhears a message, it appends the running counter value to the message, prior to inserting it into its local Bloom filter. This step essentially preserves in the witnesses' local fingerprint the chronological sequence of the messages as recorded in the gateway log.

To prevent retroactive data tampering, the individual witness records are also chained together with hash values of the previous witness record and of the corresponding data block in the gateway log. The structure of the witness record is as depicted in Fig. 2. The witnesses also track the number of messages, n, inserted in their Bloom filters and include the count in the record. Each record is digitally signed by the witness (thereby enabling source authentication and non-repudiation) and then sent to the medical server.

3.2.3 Verification

In the event of an incident, forensics specialists can use these witness records to verify the integrity of the gateway log. The investigators would take individual messages from the gateway log, append the counter value, and pass them through the same set of hash functions used by the witness. If the output matches with that of the Bloom filter in the witness record, the log entry is genuine with very high probability (depending on the Bloom filter's parameters).

Given that packet loss is inevitable in wireless networks, all witnesses will not overhear all communications. However, as our experimental results (in §5) indicate, there is a very high probability that a message is overheard by at least one witness. We may have strong confidence in the gateway log if it is supported by the witness records. However, there is a certain probability of false positives inherent in Bloom filters. The verifying party can calculate the probability of false positives for a witness in an epoch using the following relation derived from Eq. 4

$$f \approx e^{-\frac{m(ln2)^2}{n}} \tag{5}$$

where n is the number of entries in the filter (as included in the witness record) and m is the size of the Bloom filter.

If a message in the log is verified in this manner by multiple witnesses, the verifier can quantify the confidence level for that message or data item by computing the complement of the product of the corresponding false positive probabilities. Mathematically, let f_p be the probability of false positive in the Bloom filter managed by the pth-witness, then the probability of false positive of the ith-packet, provided independent hash functions used in the Bloom filters (see §4.1), is given by:

$$F_i = \prod_{p=0}^{w_i} f_p \tag{6}$$

Here, we define $f_0 = 1$ when $w_i = 0$, i.e. the probability of false positives is 1 when there are no witnesses to log the packet.

We define the confidence level τ in the validity of ith-packet as below:

$$\tau_i = 1 - F_i = 1 - \prod_{p=0}^{w_i} f_p \tag{7}$$

We consider an example as follows: if a packet has been witnessed by none of the witnesses, the confidence level is zero. If a packet has been verified by only one witness with a probability of false positive 20%, the confidence level in the validity of that packet equates to 80%. However, If three witnesses verify a packet with the same probability of false positive 20% each, the cumulative probability of false positive becomes 0.8% and the confidence level is found to be 99.2%.

We discuss our solution's Bloom filter parameters and the inherent tradeoffs involved in greater detail in §4.

3.3 Security Analysis

Our scheme enables a number of important properties. For one, our solution relies heavily on symmetric key primitives on the witness devices to ensure the mechanism is lightweight and suitable for resource-constrained devices. Digital signatures are only used at the conclusion of an epoch to sign the data blocks and the witness records.

The use of digital signatures on the epoch-level blocks of gateway logs and witness records ensures integrity and authenticates both medical data logged by the gateway and the witness records. Signatures also ensure non-repudiation on the part of the signers. Moreover, the data cannot be altered by any other entity without compromising the private keys of the gateways and the witnesses.

Chaining of epoch-level blocks enables chronological ordering of blocks and prevents any form of retroactive data tampering in both medical data and the witness records. Even if an attacker were to compromise keys for the gateway or the witnesses, it would be exceedingly difficult for him to alter data blocks or witness records in the past since the data is chained together using hash fingerprints of prior blocks and records. Any retroactive tampering will break the chain and can be easily detected. Moreover, the logs and witness records are replicated in more than one location and any change at one site can be detected at the others.

Our use of witnesses to log the fingerprint of medical data introduces some interesting security properties. It not only secures the chronological ordering and the integrity of individual data items but also provides a loose form of localization or proximity. It also secures the contextual relationship between various data sources which may play

2. Devices that are not present at the start of the epoch may synchronize their counters using two way message exchange with the gateway as in Timing-Sync Protocol for Sensor Networks (TPSN) [44].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2927674, IEEE Transactions on Dependable and Secure Computing

a vital role in a medical practitioner's diagnosis by examination of physiological and environmental factors affecting the patient reported by other sensors in the vicinity (as discussed in §2.1).

In case a malicious gateway alters the contents of a message it has received by the wearable device, there is a very high chance that the change would be detected against the witness record. This will result in a very low confidence level in the validity of the message rendering the attack ineffective. Altering just the timestamp of a message (by changing its position in the overall chronological ordering) will have a similar effect since witnesses append running counter values (indicative of time) to each message before inserting them in the Bloom filter.

The situation is much the same for the case of sensing devices that may be hacked to act dishonestly. If a device misbehaves, its records will clash with those of the gateway and with other witnesses. Moreover, the forensic investigator can discredit any backfilled medical data as the witnesses' records will be showing it in the future epochs.

If the medical or insurance server or both are compromised and the blame is placed on the patient for tampering with the gateway log, the witnesses' records can help assign liabilities. Such a case happened in 2016 where an Australian health insurer, CommInsure, colluded with the doctors to tamper the medical records of the patients to reject their claims [17].

Regarding collusion attacks, except for witnesses, if all the entities namely sensor (retroactively), gateway, medical and insurance servers collude, the counter testimony from potential witnesses will lower the trust in such a data. The strong protection afforded by our solution derives from the fact that records are maintained independently by each party in the system. For an attacker to successfully defeat our system, he has to hack into each participating device and obtain its credentials.

The witness record can also give forensics personnel insight into the location of the device or patient. If a user makes a claim about his location at a certain time, the witness record at the location can be used to check his claim. This may be useful to detect instances where users may put their devices on other users for dishonest purposes (such as fake medical conditions, claim health benefits, etc).

3.4 Adaptation to Heterogeneous Environment

In this section, we speculate on the practical concern of how our protocol may adapt to various deployment scenarios such as hospital and home, as well as incorporate heterogeneous devices from multiple vendors with varying capabilities.

In a smart hospital, it is more likely that the sensor devices are custom-made to support the protocol. Sensing devices with an appropriate wireless communication technology such as Bluetooth, WiFi, or IEEE 802.15.6, broadcast the readings, which are collected by their respective gateways installed in the hospital building. Sensors in the same broadcast domain act as witnesses to each other. The witness records are sent to the central medical server through respective gateways. Neighboring gateways can witness sensor data as well if in the same broadcast domain.

In a home or office scenario, smart devices deployment is very likely to be heterogeneous in nature. Some manufacturers would not allow their devices to communicate and forward data to third party servers. Witness records from such devices can be forwarded to their manufacturer servers and the forensic expert can access the witness records through manufacturers when needed. The popular wireless technologies used in this scenario are Bluetooth, Zigbee (IEEE 802.15.4), and IEEE 802.15.6. Most of the popular smart home controllers in the market today, such as Google Home and Amazon Echo, support Bluetooth already except HomePod from Apple that only supports WiFi. The newly released Amazon Echo Plus supports Zigbee too. The sensors and other smart devices listening to the same wireless channel as the medical sensor will witness the data in this scenario.

We believe that our solution could be implemented to the devices in the form of a software patch and does not involve too much effort, time, and cost.

4 Design Parameters

In this section, we discuss the design choices for our scheme including the selection of hash functions, their optimal number, the size of the Bloom filter, and probabilistic logging parameter.

4.1 Hash Functions in Our Scheme

Hash functions play a fundamental role in Bloom filters and are chosen carefully depending upon the application. Popular choices of hash functions include MD5, Murmur3, CRC32, and SHA1. The following factors need to be and have been considered while selecting hash functions in our scheme:

Uniformity: As a fundamental requirement, hash function used in a Bloom filter must have its output uniformly distributed over the length of the Bloom filter.

Security: Whether to choose a cryptographic or non-cryptographic hash function depends upon the application. In our scheme, 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.

Cost: As wearable devices are resource-constrained, we need to reduce the computation cost of the hash function used. *Double hashing technique* [45] 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.

Independence: In our scheme, confidence 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 Bloom Filter Size

As apparent from Eq. 4, the Bloom filter size depends upon number of elements to be inserted as well as the probability of false positive (relates to the targeted accuracy of verification). Moreover, the filter size and number of elements

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2927674, IEEE Transactions on Dependable and Secure Computing

inserted determine the optimum number of hash functions used, as given in Eq. 3.

Studies find that the size of an acute care unit (ACU) of most hospitals vary from 10 to 100 beds while recommended ICU size is 8–12 beds [46]. Assume a witness device that manages a Bloom filter is in an ICU that holds 10 such patients (beds). Consider three vital signs (axillary temperature, heart rate, and respiration rate) being monitored from each patient in the ICU and transmitted to the gateway every 2 minutes (rate used by a medically approved device SensiumVitals System [47]). The cumulative transmission rate from a patient with three devices becomes 0.025 pkts/sec. The rate at which the witness overhears the messages becomes 0.25 pkts/sec. In cases where there is more frequent reporting (e.g. ECG) or multiple sensors, however, this rate can reach or exceed 1 pkt/sec. Now the number of elements inserted in the Bloom filter will depend upon the reception rate as well as the epoch size. Recall that an *epoch* is a time period after which witness devices forward their signed data (Bloom filters) to the gateway. Table in Fig. 3 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.

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

Fig. 3. Bloom filter design choices for a hospital ward

It is apparent from the table in Fig. 3 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 of Bloom filters. Hence, Bloom filter size is decided by estimating the density of sensors and their typical transmission rate in the vicinity where they are to be deployed.

The degree of impact this additional data communication has on witnesses is relative and compares to the data transmission rate due to their primary function. For example, an IEEE 802.15.4 compliant ECG sensor node [48] has a data rate of 12kbps, which translates to approximately 5400KB data transmitted every hour, in which case the witnessed data has little impact on the witness device. In case a witness device's data rate is around 5bps, which is extremely low, only then the witness data is comparably significant, however, it still depends upon the power capacity of the witness device.

4.3 Probabilistic Witnessing

As mentioned earlier that where there is dense deployment of sensors, probabilistic witnessing may be adopted. To enable probabilistic witnessing, the sensing device needs to append the number of potential witnesses to each packet it broadcasts. The estimate can be done based on the wireless communication it overhears, which it keeps updating. Let W_i be the number of potential witnesses the sensor device includes in the ith-packet. The witnesses then log the packets with a probability depending upon the number W_i by reading it from the packet. The probability with which a witness logs the ith-packet is given by:

$$\rho_i = \begin{cases} \frac{\beta}{W_i}, & \text{if } W_i > \beta \\ 1, & \text{otherwise} \end{cases} \tag{8}$$

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:

$$w_i = \begin{cases} \beta \text{ (on average)}, & \text{if } W_i > \beta \\ W_i, & \text{otherwise} \end{cases} \tag{9}$$

If $\rho_i = 1 \ \forall i$, this we refer to as **Absolute Witnessing** compared to **Probabilistic Witnessing** in which $\rho_i < 1$ for certain i in an epoch.

Recalling Eq. 7, intuitively it may seem that the confidence level in Probabilisitic Witnessing is lower than the one in Absolute Witnessing (equal at best) due to less number of witnesses. In order to study the effect of probabilistic witnessing on confidence level, lets consider a simple case scenario in which there are W witnesses to all n packets of an epoch from a sensor node. If $\rho_i = \rho = \beta/W \ \forall i$ be the probability with which witnesses log the data, ρn is the number of packets logged by each of ρW witnesses on average. Let m be the size of the Bloom filters managed by the witnesses, probability of false positive of the pth-witness, using Eq. 5, can be written as:

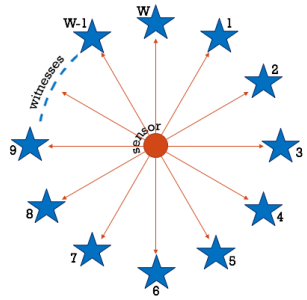

Fig. 4. Sensor data logged by W witnesses probabilistically

$$f_p \approx e^{-\frac{m(ln2)^2}{\rho n}} \tag{10}$$

The cumulative probability of false positive (using Eq. 6) for the ith-packet from ρW witnesses yields:

$$F_i = \prod_{p=0}^{w_i} f_p \approx \left(e^{-\frac{m(ln2)^2}{\rho n}} \right)^{\rho W} \tag{11}$$

This leads to the confidence level (refer to Eq. 7) as follows:

$$\tau_i \approx 1 - e^{-\frac{mW(ln2)^2}{n}} \tag{12}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2927674, IEEE Transactions on Dependable and Secure Computing

Which is same as having W witnesses to a packet. This shows that the confidence level is independent of the logging probability. It is because when the logging probability is higher, more witnesses log the data and each witness logs more packets, which increases the probability of false positive in the Bloom filter of an individual witness, however the cumulative probability of false positive decreases due to their product. Similarly, when the logging probability is low, less witnesses log the data and each witness logs less packets, which decreases the probability of false positive in the Bloom filter of an individual witness, and the cumulative probability of false positive of less witnesses equates that of more witnesses. However, low logging probability may result in verifying less number of packets which is studied later in §5.3.

5 Experiments and Results

5.1 Experiment for an Office Environment

We implemented our scheme for witness statement generation using IEEE 802.15.4 compliant 2.4GHz MicaZ wireless motes. These RF transceivers are programmable and are based on open source TinyOS operating system. They are used to develop custom sensor applications for low-power wireless sensor networks. They also work as plug and play with Crossbow's sensor boards. We conducted experiments with a human subject wearing a MicaZ mote on his right arm while walking along an office hall with cubicles (as depicted in Fig. 5). The wearable mote generated packets once-per-second, emulating an ECG heart monitor, and transmitted them at maximum power. A gateway received and logged the medical data, while three other motes were acting as witnesses – two of them were stationary and placed in adjacent cubicles, while the third was mobile and worn by another subject. The witness statements were generated using Bloom filters with the Murmur3 hash algorithm with number of hash functions $k = 7$. The filter size was set to m = 2.1 KB, chosen using Eq. 4 so as to achieve a confidence level of 99% over an epoch of half-an-hour in which up to $n = 1800$ packets may be witnessed.

We verify the packets logged by the gateway by querying the witness' Bloom filters. We find out that 97% packets are verified by at least one witness, as depicted in Fig. 6. This implementation and experimentation validates the practical feasibility of our scheme, and demonstrates its effectiveness in a small setting. We next expand the evaluation to the setting of a large hospital using real data taken from a multi-storey building.

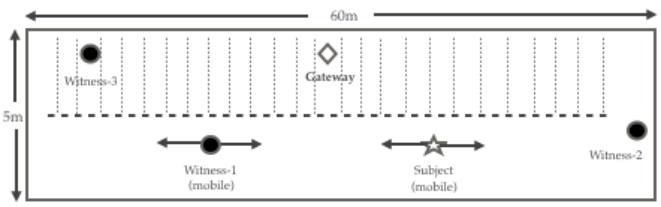

Fig. 5. Layout of the experimental setup

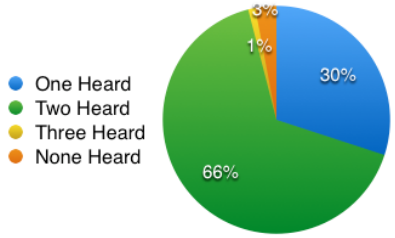

Fig. 6. Packets logged by the witnesses

5.2 Experiment for a Hospital Environment

Consider the scenario of a multi-storey hospital with multiple acute care units (ACUs) with dozens of patients in each of them. The patients' vital signs are monitored by wearable 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.

For our experiment, we chose a six-storey university building full of staff and students with their smart devices connected to various wireless access points in the building to represent the scenario of a smart hospital described above. The staff and students sitting in their office or classrooms with stationary devices may represent the patients on their beds, while the ones who are mobile may represent the patients engaged in physiotherapy sessions. The building we experimented in consists of 6 levels (Lower Ground, Ground, and Levels 1-4) with 30 wireless access points (APs) from Cisco. With the root access to the APs, we were able to collect a comprehensive set of statistics on all wireless client connections in the building for a whole day. We divided a day into 24 epochs (1 hour epoch each). The traces show that there are on average 400 clients connected to APs in an epoch. Fig. 7 shows a bar graph of average number of clients connected to each of the 30 APs in an epoch on a weekday. Additionally, the error bars show the standard deviation of the number of clients connected to these APs. There are on average 401 clients in an epoch and around 13 clients on average per AP per epoch. However, it is apparent from the graph that the load of the APs is not uniformly distributed with lower ground APs having the highest number of connections. In a hospital this can represent a ward containing patients with more sensing devices attached to their bodies.

A wireless client and the corresponding connected AP can be thought to be representatives of a sensor device worn by a patient and the gateway respectively. The rest of the APs are the potential witnesses. In order to find the probability of witnessing data to and/or from each AP by all APs in the building, we measured the coverage area of each AP in the building (corroborating it against the coverage maps provided by IT services), and determined the overlap in coverage for every pair of APs. Using these, we deduced the probability with which a client's data can be witnessed by other APs in the building given that it is connected to a certain AP. Note that the coverage area is purely required for simulation for performance purposes. A real system does not need to know the coverages.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2927674, IEEE Transactions on Dependable and Secure Computing

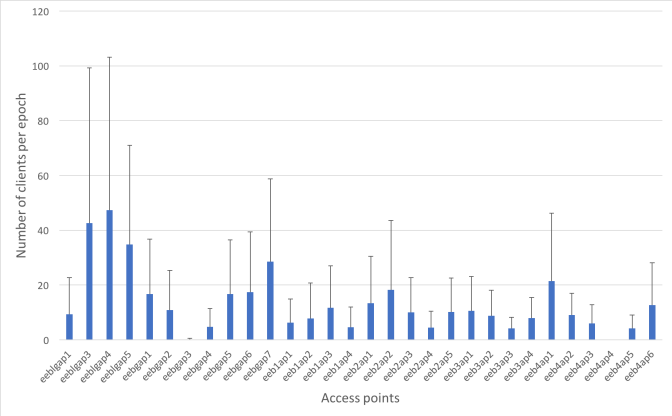

Fig. 7. Average number of clients per AP per epoch

The probability distribution of the number of AP witnesses available to a client at any time is plotted in Fig. 8, and shows over half the clients can see 4 or more AP witnesses with average number of APs visible to a client to be 4. Moreover, the average number of witnesses a client

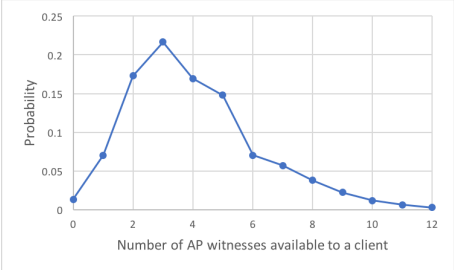

Fig. 8. Distribution of AP witnesses to a client

sees when connected to a given AP (gateway) is plotted in Fig. 9. The graph shows a maximum of 7 witnesses available to some APs on the ground floor with the average of 4. Furthermore, the probability of having a given number of

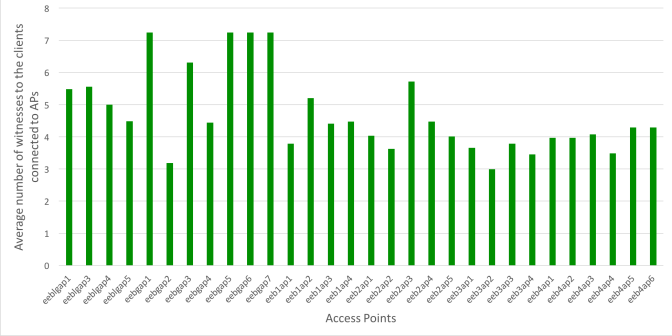

Fig. 9. Average number of witnesses to the clients connected to a given access point

witnesses for a client connected to a given AP (gateway) is plotted in Fig. 10. The graph shows that most of the APs at the ground floor have the highest probability of having more than six witnesses with 3, 4, and 5 witnesses having the highest probabilities in other APs. These plots show that adequate number of witnesses would be available in a hospital to verify medical data which we prove in the following.

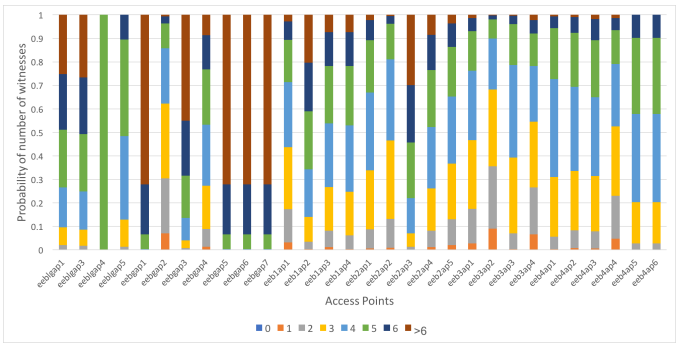

Fig. 10. Probability distribution of witnesses at each access point

We consider each client as a patient transmitting three vital signs (axillary temperature, heart rate, and respiration rate) to the gateway every 2 minutes with cumulative rate of 0.025 pkts/sec. We took the connectivity information from the traces and fed it to our implementation of the scheme while assuming the above mentioned rate. We implement the Bloom filters for each AP, generate the packets from each client and insert them to the Bloom filters managed by the witness APs. We start off with the Bloom filter size of 1KB for each AP, verify the packets, and repeat the process for a bigger size of the Bloom filter until the verification results become stable and minimize the false positives inherent to the Bloom filters. Fig. 11 shows the percentage of packets verified by at least one, two, and three witness APs against the varying Bloom filter size for each AP. It is apparent from the Fig. 11 that at a filter size of 10KB, the results converge to the true values minimizing the false positives in the verification process. It is noted that 98.7% packets are heard by at least one witness and hence only 1.3% packets remain unwitnessed.

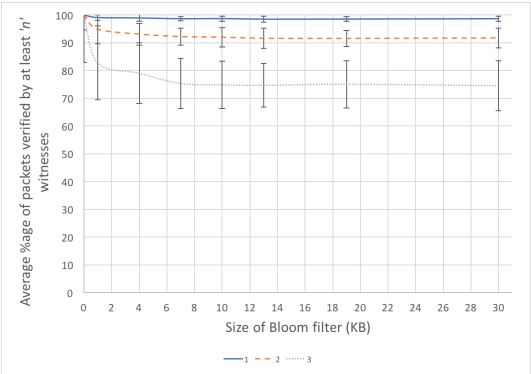

Fig. 11. Determination of the size of Bloom filter by finding convergence of packets verified by the witnesses

It is important to note that the non-uniform distribution of clients (patients) in the building (hospital) has caused some APs to witness a lot more packets than others and have contributed to the overall false positive verifications at the lower Bloom filter sizes. Had it been a uniform density of clients in the building, there would have been a significant reduction in the Bloom filter size. This problem can be mitigated by setting bigger filter sizes in the areas with large client density. Moreover, the use of Probabilistic Witnessing, as mentioned in section 4.3, can further reduce the filter size.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2019.2927674, IEEE Transactions on Dependable and Secure Computing

5.3 Probabilistic Witnessing Results

Although we propose probabilistic witnessing in dense sensor deployment environments, in order to study its effects, we implement it in our hospital environment scenario (a sparse sensor deployment facility) with 10KB Bloom filter size and five Murmur3 hash functions. We log the sensing data of an epoch (3:00pm to 4:00pm) for different values of Probabilistic Logging Parameter $(0 \leq \beta \leq 12)$, which translates to the probability of logging (refer to Eq. 8). Graph in Fig. 12 shows the logging cost of sensing data (compared to the once with absolute witnessing), the packets verified by at least one witness, as well as the confidence level (refer to Eq. 7) averaged out over all packets in an epoch for different values of Probabilistic Logging Parameter (β).

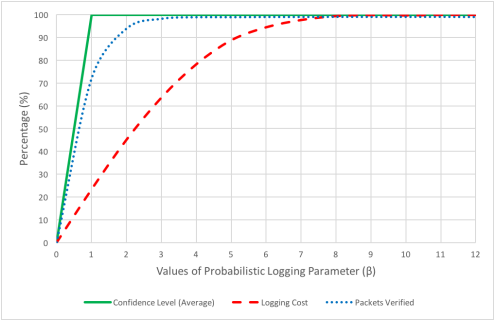

Fig. 12. Logging cost, verified packets, and confidence level against the value of probabilistic logging parameter

It is apparent from Fig. 12 that the confidence level τ is independent of Probabilistic Logging Parameter (β) as analytically shown in §4.3. Probabilistic Witnessing offers a much more cost effective solution than that of Absolute Witnessing scheme even in a sparse sensor deployment scenario. In fact, by setting Probabilistic Logging Parameter (β) value to 3 we still get 98.2% packets verified (compared to 98.7% in absolute witnessing scheme) with a huge 37% savings in the cost without sacrificing confidence level (remains 99.9%). It not only saves us the cost of logging but also the memory used by the witnesses as 37% less packets logged translate to 37% less Bloom filter size (refer to Eq.4), hence $\beta = 3$ offers a desirable alternative in our hospital scenario.

6 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 require assurances on the authenticity, integrity, and contextual correctness of data from these devices. To meet these requirements, we presented an opportunistic 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 hash-chain fashion. We motivated our idea with promising results from our experiments with real wireless devices. We discussed design and performance parameters of our protocol and illustrated their cost benefit trade-offs through simulation results. Our scheme is the first step towards secure logging enabling forensics of medical data for wearable sensing devices.

References

[1] D. Kessler, "Google Fit app released to track all your moves," http://www.androidcentral.com/google-fit-app-released-track-all-your-moves, Oct. 2014, [Online; accessed 13-June-2016].

[2] J. Stone, "Google Smart Lens Project Finds New Partner In Novartis," http://www.ibtimes.com/google-smart-lens-project-finds-new-partner-novartis-1628484, Jul. 2014, [Online; accessed 12-June-2016].

[3] D. F. Carr, "Apple Partners With Epic, Mayo Clinic For HealthKit," http://www.informationweek.com/healthcare/mobile-and-wireless/apple-partners-with-epic-mayo-clinic-for-healthkit/d/d-id/1269371, Mar. 2014, [Online; accessed 12-June-2016].

[4] R. Lawler, "Samsung Opens Simband Applications And SAMI Data Exchange Platform To Developers," http://techcrunch.com/2014/11/12/samsung-digital-health-developer-tools/, Nov. 2014, [Online; accessed 13-June-2016].

[5] PwC, "The Wearable Future," http://www.pwc.com/us/en/technology/publications/assets/pwc-wearable-tech-design-oct-8th.pdf, 2015, [Online; accessed 13-June-2016].

[6] Wearable, "Welcome to the 100 million club: Wearable sales soar in 2016," https://www.wareable.com/wearable-tech/welcome-to-the-100-million-club-wearable-sales-soar-in-2016-2852, [Online; accessed 04-Feb-2017].

[7] "Top 3 areas for wearable medical devices. source: Future Markets Insights," http://www.todaysmedicaldevelopments.com/article/wearable-medical-devices-2016-market-72816/, [Online; accessed 25-Sept-2017].

[8] N. Ziady, "Telehealth reduces hospital mortality rates," http://www.healthcarecommunication.com/Main/Articles/Telehealth_reduces_hospital_mortality_rates_9047.aspx, Oct. 2012, [Online; accessed 12-June-2016].

[9] J. H. Insurance, "John Hancock Introduces a Whole New Approach to Life Insurance in the U.S. That Rewards Customers for Healthy Living," http://www.johnhancock.com/about/news_details.php?fn=apr0815-text&yr=2015, Apr. 8, 2015, [Online; accessed 13-June-2016].

[10] UnitedHealthcare, "UnitedHealthcare and Qualcomm Collaborate to Launch New Wellness Program That Links Financial Incentives with the Use of Wearable Devices," http://www.unitedhealthgroup.com/Newsroom/Articles/Feed/UnitedHealthcare/2016/0301QualcommUnitedHealthcareMotion.aspx?r=1, March 1, 2016, [Online; accessed 13-June-2016].

[11] 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," http://www.prnewswire.com/news-releases/national-australia-banks-mlc-on-track-program-leverages-big-cloud-analytics-predictive-analytics-from-wearable-devices-and-internet-of-things-data-300228615.html, March 1, 2016, [Online; accessed 13-June-2016].

[12] D. Halperin, T. S. Heydt-Benjamin, B. Ransford, S. S. Clark, B. Defend, W. Morgan, K. Fu, T. Kohno, and W. H. Maisel, "Pacemakers and implantable cardiac defibrillators: Software radio attacks and zero-power defenses," in *Proceedings of the 2008 IEEE Symposium on Security and Privacy*, ser. SP '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 129–142.

[13] A. Hern, "Hacking risk leads to recall of 500,000 pacemakers due to patient death fears," https://www.theguardian.com/technology/2017/aug/31/hacking-risk-recall-pacemakers-patient-death-fears-fda-firmware-update, Aug. 2017, [Online; accessed 04-Sept-2017].

[14] C. Li, A. Raghunathan, and N. K. Jha, "Hijacking an insulin pump: Security attacks and defenses for a diabetes therapy system," in *2011 IEEE 13th International Conference on e-Health Networking, Applications and Services*, June 2011, pp. 150–156.

[15] J. Finkle, "J&J warns diabetic patients: Insulin pump vulnerable to hacking," http://www.reuters.com/article/us-johnson-johnson-cyber-insulin-pumps-e-idUSKCN12411L, Oct. 4, 2016, [Online; accessed 13-June-2016].

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

[17] R. Fogarty, "CommInsure: Who's who in the Commonwealth Bank's life insurance scandal?" http://www.abc.net.au/

news/2016-03-07/comminsure-scandal-whos-who-four-corners/7226576, Mar. 2016, [Online; accessed 13-June-2016].
[18] 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.
[19] B. Shebaro, S. Sultana, S. Reddy Gopavaram, and E. Bertino, "Demonstrating a lightweight data provenance for sensor networks," in *Proceedings of the 2012 ACM conference on Computer and communications security*. ACM, 2012, pp. 1022–1024.
[20] icontrol Networks, "2015 State of the Smart Home Report," https://www.icontrol.com/wp-content/uploads/2015/06/Smart_Home_Report_2015.pdf, 2015, [Online; accessed 13-June-2016].
[21] M. Rushanan, A. D. Rubin, D. F. Kune, and C. M. Swanson, "Sok: Security and privacy in implantable medical devices and body area networks," in *2014 IEEE Symposium on Security and Privacy*. IEEE, 2014, pp. 524–539.
[22] K. Kent, S. Chevalier, T. Grance, and H. Dang, "Guide to integrating forensic techniques into incident response," *NIST Special Publication*, vol. 10, pp. 800–86, 2006.
[23] A. Prasad, R. A. Peterson, S. Mare, J. Sorber, K. Paul, and D. Kotz, "A provenance framework for mhealth." in *COMSNETS*. IEEE, 2013, pp. 1–6. [Online]. Available: http://dblp.uni-trier.de/db/conf/comsnets/comsnets2013.html#PrasadPMSPK13
[24] S. Berinato, "The Rise of Anti-Forensics," https://www.csoonline.com/article/2122329/investigations-forensics/the-rise-of-anti-forensics.html, Jun. 2007, [Online; accessed 12-June-2016].
[25] C. Hauser, "In Connecticut Murder Case, a Fitbit Is a Silent Witness," https://www.nytimes.com/2017/04/27/nyregion/in-connecticut-murder-case-a-fitbit-is-a-silent-witness.html?mcubz=1, Apr. 2017, [Online; accessed 12-June-2016].
[26] H. Fereidooni, J. Classen, T. Spink, P. Patras, M. Miettinen, A.-R. Sadeghi, M. Hollick, and M. Conti, "Breaking fitness records without moving: Reverse engineering and spoofing fitbit," *arXiv preprint arXiv:1706.09165*, 2017.
[27] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, pp. 422–426, 1970.
[28] A. Broder and M. Mitzenmacher, "Network Applications of Bloom Filters: A Survey," *Internet Mathematics*, vol. 1, no. 4, pp. 485–509, 2003.
[29] 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.
[30] R. Accorsi, "Safe-keeping digital evidence with secure logging protocols: State of the art and challenges," in *IT Security Incident Management and IT Forensics, 2009. IMF'09. Fifth International Conference on*. IEEE, 2009, pp. 94–110.
[31] B. Schneier and J. Kelsey, "Secure audit logs to support computer forensics," *ACM Trans. Inf. Syst. Secur.*, vol. 2, no. 2, pp. 159–176, May 1999.
[32] J. E. Holt, "Logcrypt: forward security and public verification for secure audit logs," in *Proc. of the Australasian workshops on Grid computing and e-research*. Darlinghurst, Australia: Australian Computer Society, Inc., 2006, pp. 203–211.
[33] R. Accorsi, *BBox: A Distributed Secure Log Architecture*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 109–124. [Online]. Available: https://doi.org/10.1007/978-3-642-22633-5_8
[34] D. Ma and G. Tsudik, "A new approach to secure logging," *Trans. Storage*, vol. 5, no. 1, pp. 2:1–2:21, Mar. 2009.
[35] B. Waters, B. R. Waters, D. Balfanz, D. Balfanz, G. Durfee, G. Durfee, D. K. Smetters, and D. K. Smetters, "Building an encrypted and searchable audit log," in *In The 11th Annual Network and Distributed System Security Symposium*, 2004.
[36] Y. Ohtaki, "Partial disclosure of searchable encrypted data with support for boolean queries." in *ARES*. IEEE Computer Society, 2008, pp. 1083–1090.
[37] 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.
[38] 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.
[39] A. De La Piedra, A. Braeken, A. Touhafi, and K. Wouters, "Secure event logging in sensor networks," *Computers & Mathematics with Applications*, vol. 65, no. 5, pp. 762–773, 2013.
[40] M. Siddiqi, S. T. Ali, and V. Sivaraman, "Secure lightweight context-driven data logging for bodyworn sensing devices," in *5th International Symposium on Digital Forensic and Security (ISDFS)*. IEEE, 2017, pp. 1–6.
[41] K. Crawford, "When Fitbit is the Expert Witness," https://www.theatlantic.com/technology/archive/2014/11/when-fitbit-is-the-expert-witness/382936/, Nov. 2014, [Online; accessed 14-Dec-2017].
[42] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.
[43] 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.
[44] S. Ganeriwal *et al.*, "Timing-sync protocol for sensor networks," in *ENSS*. New York, NY, USA: ACM, 2003.
[45] A. Kirsch and M. Mitzenmacher, "Less hashing, same performance: Building a better bloom filter," *Random Struct. Algorithms*, vol. 33, no. 2, pp. 187–218, Sep. 2008.
[46] 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.
[47] Sensium, "SensiumVitals System: Wireless monitoring of vital signs," https://www.sensium-healthcare.com/sensiumvitals%C2%AE-system#.WG5PdbZ94Xp, [Online; accessed 06-Jan-2017].
[48] X. Liang and I. Balasingham, "Performance analysis of the ieee 802.15. 4 based ecg monitoring network," in *Proceedings of the 7th IASTED international conferences on wireless and optical communications*, 2007, pp. 99–104.

Muhammad Siddiqi obtained his degree in electronics from Quaid-i-Azam University, Pakistan, in 2004. He worked in ZTE Corporation at Nanjing R&D Centre in China from 2005 to 2009. He received his master's degree in telecommunications in 2012 from the University of New South Wales, Australia, where he is currently pursuing his PhD degree in electrical engineering. His research interests include body area networks, the Internet of Things, and network security.

Syed Taha Ali did his BSc. (Eng) from GIK Institute of Engineering Sciences and Technology, Pakistan in 2002. He did his MS and PhD in Electrical Engineering from the University of New South Wales, Australia, in 2006 and 2012. He did his postdoctoral research in the University of New South Wales, Australia, in 2013 and in Newcastle University, UK, from 2014 to 2016. He is now Assistant Professor at National University of Science and Technology, Pakistan. His research interests include body area networks, software defined networking, network security, and crypto-currencies.

Vijay Sivaraman received his B. Tech. from the Indian Institute of Technology in Delhi, India, in 1994, his M.S. from North Carolina State University in 1996, and his Ph.D. from the University of California, Los Angeles in 2000. He has worked at Bell-Labs and a Silicon Valley startup manufacturing optical switch-routers. He is now Professor at the University of New South Wales, Australia. His research interests include software defined networking and IoT technologies.