Authentication of Lossy Data in Body-Sensor Networks for Cloud-based Healthcare Monitoring[☆]

Syed Taha Ali[a,*], Vijay Sivaraman[a], Diethelm Ostry[b]

[a] *University of New South Wales, Sydney, Australia*
[b] *ICT Centre, Commonwealth Scientific and Industrial Research Organisation (CSIRO), Australia*

Abstract

Growing pressures on healthcare costs are spurring development of lightweight bodyworn sensors for real-time and continuous physiological monitoring. Data from these sensors is streamed wirelessly to a handheld device such as a smartphone or tablet, and then archived in the cloud by personal health record services. Authenticating the data these devices generate is vital to ensure proper diagnosis, traceability, and validation of claims. Digital signatures at the packet-level are too resource-intensive for bodyworn devices, while block-level signatures are not robust to loss. In this paper we propose, analyse, and validate a practical, lightweight robust authentication scheme suitable for health-monitoring. We make three specific contributions: (a) We develop an authentication scheme that is both low-cost (using a Merkle hash tree to amortise digital signature costs), and loss-resilient (using network coding to recover strategic nodes within the tree). (b) We design a framework for optimising placement of network coding within the tree to maximise data verifiability for a given overhead and loss environment. (c) We validate our scheme using experimental traces of typical operating conditions to show that it achieves high success (over 99% of the medical data can be authenticated) at very low overheads (as low as 5% extra transmissions) and at very low cost (the bodyworn device has to perform a digital signature operation no more than once per hour). We believe our novel authentication scheme can be a key step in the integration of wearable medical monitoring devices into current cloud-based healthcare systems.

Keywords:
body sensor networks, data authentication, hash trees, network coding

1. Introduction

Increase in age-related disabilities and chronic medical conditions is putting huge pressure on national health expenditures worldwide. The US spends $2.3 trillion, or 16% of

[☆]This submission is an extended version of a paper presented at the IEEE International Conference on Sensing, Communication, and Networking (SECON), Seoul, June, 2012

*Corresponding author: Syed Taha Ali, School of Electrical Engineering and Telecommunications, UNSW, Sydney, NSW 2052, Australia. Email: taha@unsw.edu.au, Tel: +61 2 9385 4477, Fax: +61 2 9385 5993.

Email addresses: taha@unsw.edu.au (Syed Taha Ali), vijay@unsw.edu.au (Vijay Sivaraman), diet.ostry@csiro.au (Diethelm Ostry)

its GDP, on healthcare, and these costs are projected to rise steeply in coming years. A promising approach to dramatically cut costs is the emerging paradigm of mobile-health, which consists of bodyworn wireless sensor nodes that interface with handheld devices (such as smartphones and tablets), enabling cloud-based continuous monitoring (and possible treatment) of patients in their homes. Wearable platforms have recently begun to appear for personalized healthcare: the Sensium Digital Plaster [26] is a bodyworn wireless solution that monitors a subject's ECG, temperature, and movement. Efforts are underway to develop sensor devices that interact with the iPhone and iPad [2] and Android devices [12]. ABI research predicts 59 million wearable home health devices will be in use by 2014 [24].

However, for wearable medical monitoring devices to be integrated into the current healthcare system, doctors need to be able to trust the data these devices generate, as do insurance companies and government agencies that provide benefits. Given the critical importance of medical data and the huge associated liabilities, there have to be iron-clad guarantees as to source and data integrity. Specifically, the data stored in the cloud should be traceable back to the originating device, it should be non-repudiable, and should not be forgeable by anyone, including authorized parties such as the patient or doctor or caregiver, etc.

Wearable devices are by definition small and light (the Sensium weighs under 10 grams), and hence severely constrained in computation, memory, communication, and battery resources. It is therefore tempting to offload the task of guaranteeing authenticity of the sensed data to the (more powerful) first-hop basestation, which may be a specialized unit, or an attachment to a multipurpose handheld device such as a smartphone. However, software on the basestation can be easily tampered with and secret keys extracted, rendering the data reported by the basestation (to a local or central database in the cloud) untrustworthy. Moreover, traceability of the medical data would only extend back to the basestation, and not to the bodyworn device, which is problematic when errors and malfunctions (which may carry heavy liabilities) need to be isolated. The data may be subject to tampering in the cloud itself by hackers and even parties authorized to legitimately access it. These requirements necessitate data authenticity be guaranteed by the source, namely the bodyworn device, rather than an intermediate transit point or destination.

Public-key cryptography is ideal for delivering conceptually simple and highly scalable at-source authentication of medical data without requiring complex key management. Data generated by the bodyworn device can be "digitally signed" by hashing the data content and encrypting with the device's private key. Any entity can authenticate the data by verifying the signature using the device's public key (which may be made publicly accessible). However, digital signatures are computationally expensive, typically two to three orders of magnitude more costly than symmetric-key operations.

In this paper we design, analyse, optimise and evaluate a novel authentication scheme whereby the bodyworn device need only perform digital signatures infrequently (e.g. once an hour, over a large block of data), thereby reducing energy costs, and the receiver can verify most of the data even in the presence of losses. We clarify here that our objective is *not* to recover lost data packets (that is deemed too complex and costly), but to be able to authenticate *received* data packets even if other packets in the data set are lost.

Our scheme leverages the idea of a Merkle *hash tree* together with *network coding*. The sender combines hashes of the data items to form a tree and digitally signs only the root. The receiver validates the data by repeated hashing along the "authentication path" till the root is reached. Due to packet loss, nodes along the authentication path may not be available to the receiver. The sender therefore applies network coding to strategically insert "recovery packets" to help the receiver reconstruct the authentication path. We show that, if configured properly, recovery packets dramatically improve authentication of the data with very low computation and transmission overheads, even in the presence of loss.

Our specific contributions are: **First**, we develop a novel low-cost scheme for authenticating lossy data by combining a Merkle hash tree (to amortise authentication cost) with strategic use of network coding (to recover lost hash nodes in the tree). **Second**, we develop an optimization framework that, for given loss conditions and specified overhead, determines a baseline for use of coding to maximize the fraction of data items that can be successfully verified. We then improve significantly upon this estimate using the Gilbert model to simulate packet loss in a bodyworn environment.**Third**, we validate our findings using experimental traces of typical operating scenarios ($\sim$2% packet loss) to show that it allows nearly all (over 99%) of the medical data to be verified in a dynamic online setting for very low cost in transmission overheads (less than 4%) and computation (a digital signature operation once per hour). To the best of our knowledge our scheme is the first to provide a practical way of ensuring authenticity of lossy data at very low energy costs, suitable to the emerging paradigm of continuous cloud-based healthcare monitoring.

The rest of this paper is organized as follows: in Section 2, we discuss the system model, prior work, and briefly introduce hash trees and network coding. We describe our solution in detail in Section 3, and formulate the optimization in Section 4. We support our findings in Section 5 with results from simulations and experiments in real settings using bodyworn devices. We conclude in Section 6.

2. System Model and Background

In this section we detail the system architecture, operating assumptions and threat model, we discuss prior work in this domain and introduce hash trees and network coding.

2.1. System Architecture

There is a marked trend in proposed deployments of biomedical sensor networks towards interconnecting and integrating various types of devices such as wireless sensors, personal digital assistants (PDAs), smartphones, tablets, PC-class systems, online databases in the cloud, etc. Examples include CEA-Leti's CORMORAN [4] project for inter-network cooperation, location and navigation; the MobiHealth [16] and the WiserBAN project [3] targeting continuous patient monitoring and value-added healthcare; COLLAGE (Collaboration on Ageing) [6] and CuPiD [28]for assisted living and residential monitoring; and, in the domain of sports, the Sesame (SEnsing for Sport and Managed Exercise) [8] consortium that collects and processes real-time data from athletes.

For our purposes, we consider a basic architecture depicted in Fig. 1, consisting of disparate devices and multiple access points. An at-home patient wears a wireless sensor

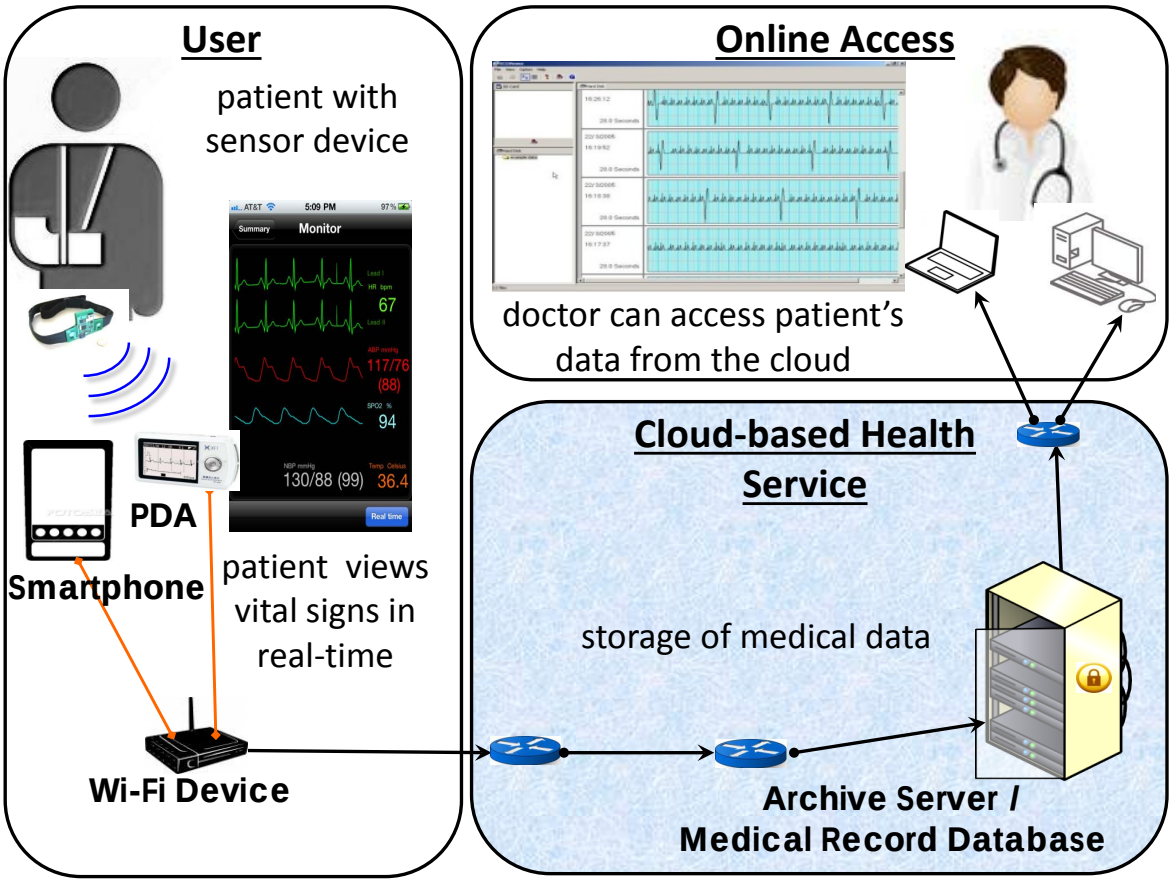

Figure 1: System Architecture

device to read his blood pressure, ECG, etc. once per second. These readings are periodically communicated to a basestation i.e. a smartphone or PDA, that may incorporate a utility to view the physiological readings, and uploads the data stream over the Internet to a centralized health service in the cloud. Authorized personnel such as doctors and nurses access the database directly to diagnose and monitor patients. The data is also of interest to litigators for forensics, to investigate malpractice, and assign liabilities. Our aim is to secure this data from tampering throughout the network while allowing it to be easily viewed and authenticated.

2.2. Operating Assumptions and Threat Model

Sensor devices have severely limited computation, memory, and communication resources. The basestation device, however, need not be restricted in resources. Packets will be dropped over the wireless channel between sensor device and basestation. Studies have shown that packet loss is very unpredictable [30] for bodyworn devices, however, it is also generally low [19] in indoor environments ($1 \sim 3\%$ as per our experiments) due to the rich multipath. Our solution is specifically targeted for such low-loss environments.

We would note here that our solution does not specifically address the issue of data privacy in body-sensor networks and cloud storage. The sanctity of the doctor-patient relationship necessitates confidentiality of patient data, for which strict access control and

accountability solutions are currently lacking. Furthermore, due to the distributed nature of the cloud, personal and medical data may be transported across geographic boundaries for storage, where different legislation may govern data handling. We believe this issue of data confidentiality is a separate research problem, distinct from that of authentication. And, we would also point out that our authentication solution is perfectly compatible with basic privacy mechanisms such as secret-key encryption. The operator of the scheme can easily configure sensor data to be encrypted on-device before it is processed by our scheme. Data integrity across the network will still be guaranteed, but now only authorized users will be able to view the data.

Our focus in this paper is on two main types of threat: first, an adversary can easily eavesdrop on messages, masquerade as another entity, and inject false data into the network. Second, is the internal threat: a genuine user of the device may seek to deliberately alter his physiological data to secure benefits by hacking into his cell phone, or an authorized user may use his credentials to log onto cloud-based archive and alter existing data. We do not consider more advanced attacks in this paper, such as denial-of-service or jamming attacks.

2.3. Related Work

In this section, we briefly discuss the shortcomings of existing approaches to authentication, and we situate our research contribution in the field of data stream authentication. Several stream authentication protocols exist in the literature, and their design concerns include authentication properties (per link or end-to-end), nature of the data stream (offline, i.e. data is known to the sender beforehand or online, i.e. data generated in real time). Typical performance metrics considered are computation cost, communication overhead, sender and receiver memory buffer size, authentication delay, and loss tolerance.

2.3.1. Message Authentication Codes

A low-cost means of guaranteeing authenticity of a data item is to generate a message authentication code (MAC), which can be verified by any party that has the shared secret key. However, the need to keep the key secret is problematic when applied to healthcare monitoring at scale because: (a) a single authority will be required for managing, and distributing keys for hundreds of thousands of bodyworn devices which gives rise to logistic problems and requires strong measures to protect against compromise (which can put the entire patient population's data at risk), (b) multiple entities (e.g. medical practitioners, insurance companies, etc.) cannot authenticate the data unless each has access to the secret key (increasing risk of compromise or creating a performance bottleneck that the health system can ill afford).

2.3.2. Digital Signatures

Digital signatures give ironclad, non-repudiable guarantees to source and integrity, but are impractical for resource-constrained devices. On the popular Mica2/MicaZ wireless sensor network platform, signature generation with RSA (using a 1024-bit key) takes about 12 seconds, consumes 360 mWs, and with ECC (using a 160-bit key) takes 0.9 seconds, consuming 27 mWs [22]. By way of comparison, this energy is two to three

orders of magnitude higher than for a hashing operation: an SHA-1 operation over a 16-byte block of data consumes about 112 μWs.

This high energy cost of a digital signature operation demands that it be performed sparingly by the bodyworn device. In other words, the body-sensor device transmits, for example, every hour a single digital signature authenticating all sensing data transmitted over that period. Amortising the cost of the digital signature thus over a large set of data saves precious energy, but has the risk that if even one piece of data is lost, the signature is rendered unverifiable, and no data in that set can be authenticated. Packet loss is inevitable in dynamic environments, and one may think this problem can be overcome by having the bodyworn device compute the signature over those data items in the block that have *successfully* been transferred to the basestation; however this approach (a) relies on link-layer acknowledgements which may not necessarily be present in the system, (b) requires lock-step synchronisation between the device and base to ensure that the signature is computed over exactly the same set of data packets, which can be problematic to implement given that loss can happen in both directions (i.e. data packets and acknowledgement packets), and (c) precludes scenarios where multiple base-stations are present (e.g. in a hospital or a disaster-recovery scenario) wherein any base can pick up the packet transmitted by the bodyworn device and upload it to the database. We therefore believe the authentication scheme should not assume lossless data transfer (particularly in a body sensor network, for which extensive experimentation has indicated that movement and changes in body orientation can induce significant loss [30, 19]), but should instead be designed to be robust to data loss.

We next describe common signature amortization techniques, how they can be used for data stream authentication, and we consider their suitability for body sensor networks.

2.3.3. Hash Chaining

Hash chaining [9] is one of the simplest techniques for digitally signing streams. For offline streams, the first packet is digitally signed, and every packet sent to the receiver has appended to it the hash of the *next* packet to follow in the stream, thereby leveraging the one-way nature of the hash function to verify the whole stream. For live data streams, the initial digital signature is essentially used to authenticate a parallel chain of public keys for one-time signatures, which in turn are used to authenticate the data packets. These techniques are not loss-tolerant and incur significant communication overhead. [21], [11] and [31] augment this basic scheme for robustness by appending to every data packet additional hash values of strategically selected data packets (adding further to communication costs) such that lost links in the received hash chain may be recovered.

The authors in [20] take a different approach to transmitting the authenticating chain. Every data packet has appended to it FEC-encoded hash values and the signature of the data set such that if there are a data packets in the set, the operator of the scheme can choose a value b where $b \leq a$, such that the signature and hash values are recoverable if the receiver gets any b out of a packets. This scheme is extended in [1] where the number of hash packets transmitted is further reduced, improving efficiency without impacting performance greatly. This is possible because the receiver itself generates certain hash values from successfully received data packets.

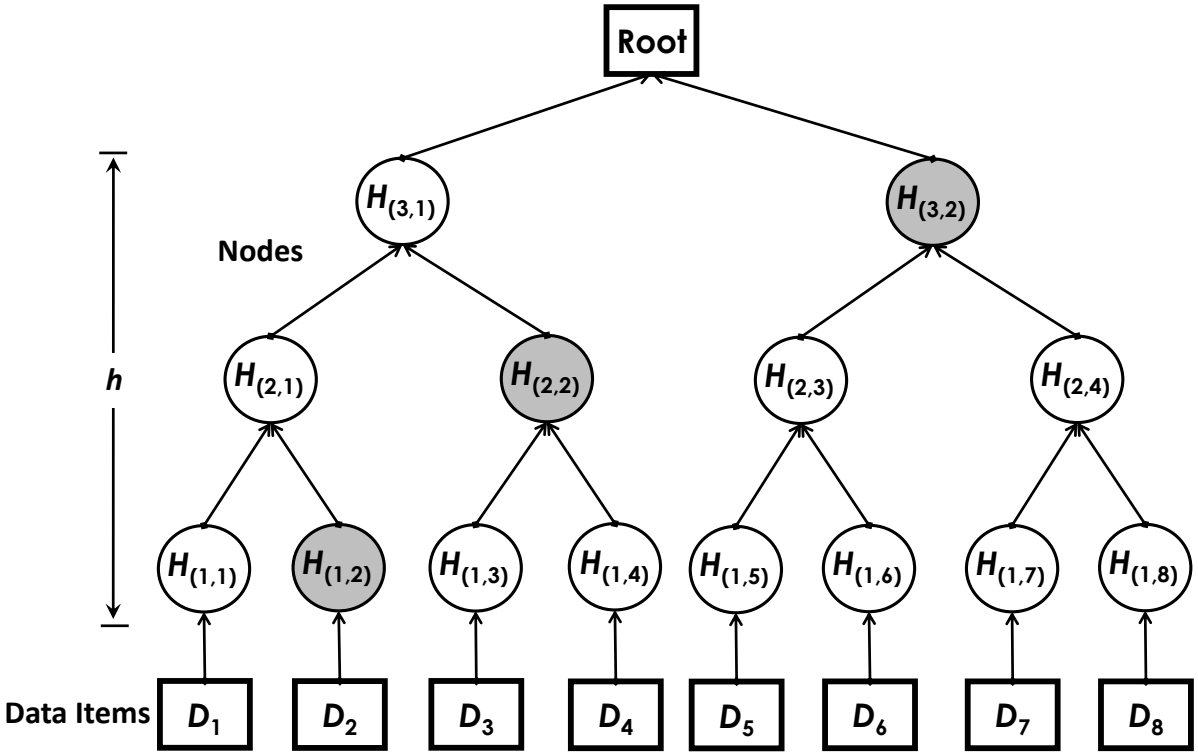

Figure 2: An 8-leaf Hash Tree

2.3.4. Hash trees

Hash trees, first proposed by Ralph Merkle [18], can similarly be used to authenticate data sets. Hash trees allow computation of a message digest over a set of data items using hash and concatenate operations to build a tree structure encompassing the entire set. A hash tree of height h is depicted in Fig. 2. Tree nodes are identified by an (i, j) tuple, such that $H_{(i,j)}$ refers to the j-th node in the i-th row (counting up from below) within the tree. The lowest level of the tree is formed by taking the hash of the corresponding data items, i.e. $Hash(D_j)$ where D_j is the j-th data item and *Hash* is a collision resilient hash function. Each internal node of the tree is computed by taking the hash of the concatenation of both its children nodes, i.e. $Hash(child_{left}|child_{right})$. Due to the one-way nature of the hash function, each node in the tree validates its children, and, by extension, their respective children, and so on, including all encompassed data items.

A digital signature on the root of the tree therefore authenticates all data items and amortizes signature cost. Assuming that the data items and signed root are reliably transmitted, the receiver reconstructs the tree from the data, verifies the signature using the sender's public key, and consequently **authenticates** all data items. This also ensures **non-repudiation** as the digital signature irrevocably commits the sender to the data. To break this mechanism, an attacker would have to break the underlying hash function (to forge nodes in the tree) or the digital signature algorithm (to forge the sender's signature).

The receiver can alternatively authenticate an individual data item in the set by receiving the item's individual **authentication path** i.e. those sibling nodes (that share

the same parent) that lie on the path from the data item to the root (shaded in Fig.2). Individual data items within the set can thus be authenticated independently. The corresponding cost is the overheads involved in transmitting the authentication path. This is the approach taken in [29] , where the authors embed the respective authentication path with each data item to verify multicasts for multimedia applications. Each data item is therefore authenticated as soon as it is received. However, this communication overhead is prohibitive for a constrained sensor device. Appending the complete authentication path for a tree of height $h = 12$, for example, results in an excessive overhead of 12x20 bytes per data item (for SHA-1).

Hash trees are also used in [13] to authenticate media files in offline P2P networks. A file is divided into data blocks and a tree built over the entire set. The signed root is communicated to the downloading client initially, which then requests peers for FEC-coded information to reconstruct the entire tree, enabling it to verify data blocks once they arrive.

[15] and [7] propose authentication for sensor network code distribution protocols (e.g. Deluge) using hash chains and trees to amortise the cost of digital signatures over entire program images. There are two differences between this approach and ours: these solutions rely on Deluge's underlying reliable transmission mechanism to ensure the data set is fully received, i.e. these solutions are not robust to loss. Second, authentication is performed over the entire image. In our case, we authenticate individual data items irrespective of other items which may be irretrievably lost.

2.3.5. Our Contribution

Our solution, designed to run on resource-constrained sensor devices, builds a hash tree over a live stream and relies on intelligent application of network coding to provide robustness to loss. Whereas most of the schemes presented in prior work appear to have one or more characteristics in common with our solution, there are key differences: compared to chaining approaches, our scheme takes advantage of the hash tree primitive, using a mere fraction of the overhead of schemes such as [20] to provide a very high degree of verifiability (experimental results show that over 99% of the data may be verified on as low as 5% extra transmissions). Another key difference is memory constraints: schemes such as [20], [13], and [29] work offline and generate coded authentication information before data transmission commences. For a bodyworn device, it is impractical to store in memory a data set comprising potentially thousands of sensor readings. Our scheme constructs a hash tree in real-time and performs network coding on tree items as soon as they are generated (after which they are discarded to conserve memory).

To the best of our knowledge, ours is the first scheme to investigate non-repudiable authentication for live lossy data streams for the unique case of severely resource-constrained devices.

2.4. Network Coding

Network coding improves network throughput and loss resilience by having a node send linear combinations of data packets, constructed so that a receiver can separate the information or reconstruct losses. We apply network coding at the packet level, which is essentially equivalent to symbol level coding with packets serving as symbols [17], to protect against packet loss between two communicating parties.

A number of coded packets (we call **recovery packets**) are constructed from the data packets. Each recovery packet X is associated with a set of n coefficients $g_1, ..., g_n$ in a finite field, and is computed symbol-wise as $X = \sum_{i=1}^{n} g_i M^i$ where $M^1, ..., M^n$ are the original data packets. To successfully recover all data, the receiver requires a sufficient number of packets from the total set of original data packets and recovery packets, i.e. if there are l data items, r recovery packets, and m total items are sent such that $m = l + r$, the receiver can reconstruct all data packets if any combination of $n \geq l$ packets are received. The imperative condition is that coefficient sets chosen for coding the recovery packets must be linearly independent. These coefficients only need to be computed once prior to deployment, so that the encoding process comprises simple finite field operations well within the computational capabilities of embedded devices [14]. Next we show how network coding can be applied to hash trees to make authentication robust to loss.

3. Our Scheme for Authenticating Lossy Data

We first describe the operation of our scheme, and then discuss its properties.

3.1. Operation of Our Scheme

The sensor device is configured with a tree height and recovery overhead. As it accumulates sensed data items, it constructs a hash tree (of configured size) on the data items as described previously, and the root is digitally signed using the the sensor's private key. Alongside, the device also constructs *recovery packets*, equal in number to the recovery overhead, that are independent linear combinations of internal nodes at a chosen level in the tree. The data items, recovery packets, and signed root are transmitted as they become available. Internal hash nodes of the tree are never transmitted.

The receiver (base-station or archival database) is assumed to reliably receive the signed root (say using a reliable transmission protocol), since verification of all data in the tree hinges upon receipt of the signature. However, data items, as well as recovery packets, are transmitted unreliably and may be lost. As data items are received, the tree is reconstructed by the receiver. Of the $N(i) = 2^{h-i+1}$ nodes at level-i in the tree of height h (levels are counted from bottom to top), say l may be received, and of the $R(i)$ recovery packets for that level, say k are received. If $l + k \geq N(i)$, the receiver can fully reconstruct tree level i, and consequently all levels above (by successive concatenation and hashing) in the tree. The verifiability of a data item is therefore no longer dependent on receiving all data items, but is reduced to the ability to reconstruct the authentication path up to level-i at which all nodes are available (via reconstruction or recovery).

A more detailed description of the operations at the bodyworn sensor device follows (refer to flowchart in Fig.3):

1. **Bootstrapping:** We assume the sensor device's public key is accessible to the receiver. Linear coefficients used for coding are pre-computed using deterministic algorithms [23] and communicated to the two communicating parties during bootstrapping, along with the allowed recovery overhead $R(i)$, and the level-i of the tree at which network coding is applied.
2. **Data Sampling, Tree Construction and Digital Signature:** Sensors collect data as per sampling frequency. Each data item may contain a number of concatenated samples. The hash tree is constructed in parallel. The data item is first

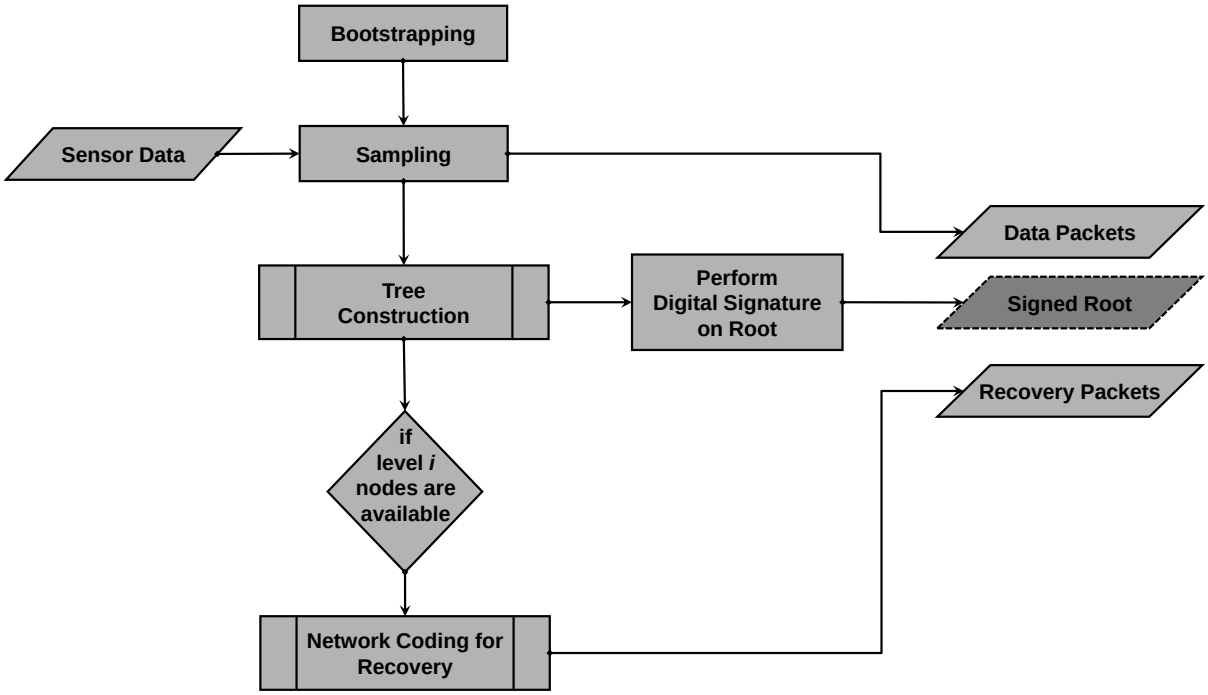

Figure 3: Process flow at sender

hashed to yield the corresponding tree leaf, then it is transmitted and deleted to conserve memory. As sibling leaves become available, they are hashed to yield the parent and then discarded. At any point in time, there are no more than h internal tree nodes buffered in the sensor device's memory and the tree exists always in a state of partial construction to minimize memory consumption, such that for a tree of height $h = 12$ with 4096 data leaves and 8191 internal hash nodes, the sender need not buffer them all, but store 1 data leaf and 12 hash nodes at any one time. When the root is computed, it is signed and transmitted using a reliable (e.g. ARQ-based) protocol. Internal hash nodes are not transmitted.

3. **Network Coding:** Coding is performed as an accumulation operation: the sensor device maintains a running buffer for $R(i)$ allowed recovery packets. As soon as any tree node is accumulated into all recovery packets it is part of, it can be discarded. When a recovery packet is ready, it is transmitted. Recovery packets are buffered in the sender's memory and therefore it is important the quantity is not too great.

Some data items and recovery packets are dropped over the wireless link. Receiver side operations (as depicted in Fig.4) are as follows:

1. **Packet Receipt, Signature Verification and Tree Reconstruction:** Received packets are mapped within the data-set. The sender's public key is used verify the signature on the root. The tree is now reconstructed bottom-up. Certain nodes in the tree will be missing due to dropped data items.
2. **Network Coding**: At level-i, received recovery packets are used to attempt recovery of missing nodes at that level. If all nodes at this level become available, upper levels of the tree can be fully reconstructed. Otherwise, upper levels will also have missing nodes.

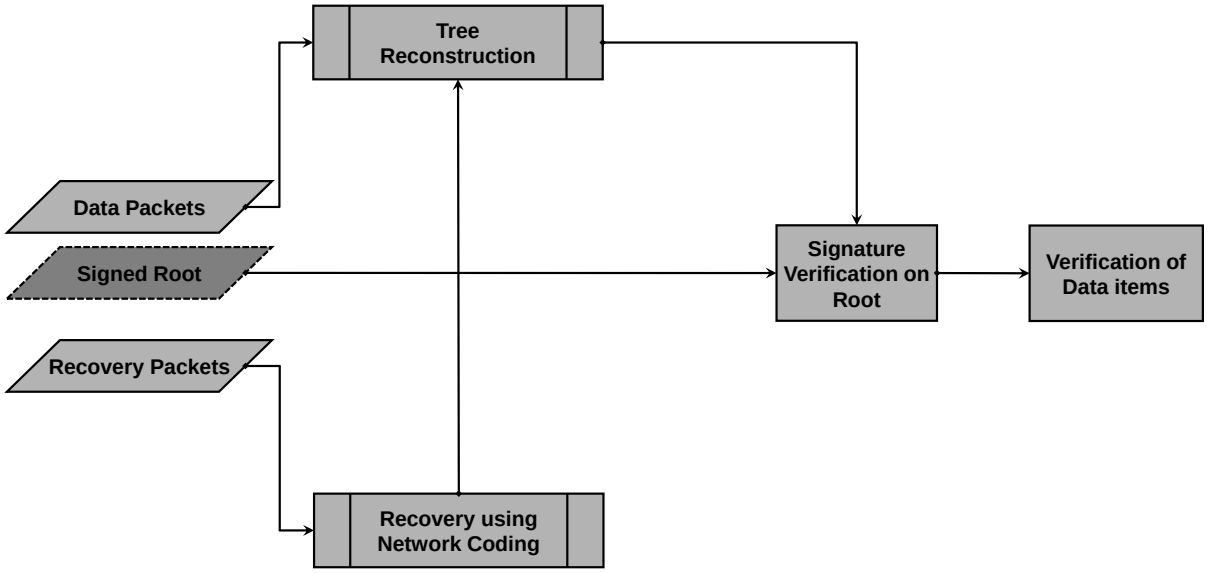

Figure 4: Process flow at receiver

3. **Data Verification:** Data items that have a complete authentication path all the way up to the root can be authenticated, whereas others cannot.

3.2. Discussion of the Scheme

We first note that we do not apply network coding on the data items themselves, since (a) our primary interest is in authenticating received data, not in recovering all lost data, (b) the data packets may be large and require unacceptable overheads for recovery whereas hashes are capped in size (e.g. 20 bytes in SHA-1), (c) the overhead for recovering missing tree nodes at a higher level can be much less than for recovering lost data items which number more, and (d) if recovery were only attempted for data items and failed for even one, all data items in the tree become unverifiable, which is catastrophic.

Our scheme guarantees authenticity and non-repudiability. Signature cost is amortised over 2^h data items in a tree of height h, and the tree structure can be configured as per operating requirements. For example, if we assume the operator of the scheme can allow a signature operation once per hour, bodyworn sensors which transmit one data item per minute (suitable for measuring heart-rate, temperature, etc.) or once per second (for ECG monitoring) can use trees of height $h = 6$ and $h = 12$ respectively. Transmission overheads are also much lower: our scheme transmits only one digital signature per tree of 2^h data items, along with a handful (typically 5-10%) of hash-digest sized coded recovery packets.

The novelty of our scheme is in applying network coding to internal tree nodes to dramatically improve authentication probability, for low overheads, in the presence of data losses. In the following section we develop a mathematical framework to determine ideal placement of network coding in the tree structure, tailored to the loss environment and transmission constraints.

4. A Framework for Optimal Placement of Network Coding

In this section, we develop a framework to determine the ideal placement of network coding within the tree to maximize the fraction of successfully authenticated data items, given loss conditions and coding overhead allowance.

We first argue (without formal proof) that for a given loss environment and given limit on (network coding based) recovery packets, it is best to apply all recovery effort to a single level of the hash tree rather than splitting it across levels. To see why, consider a case where $R(i)$ recovery packets are sent at level-i and $R(j)$ at a higher level-j in the tree (i.e., $j > i$). At the receiver, if the number of missing nodes (i.e. nodes that could not be reconstructed from their children) at level-i is no more than $R(i)$, all missing nodes can be recovered. This allows all upper levels to be reconstructed fully, and the recovery packets at level-j are thus wasted. If on the other hand the number of missing nodes at level-i exceeds $R(i)$, network coding at level-i cannot recover any missing nodes, and is thus wasted. Either way, having recovery packets at both levels is not helpful, since one of them is always wasted (this observation was also validated by our simulations). In what follows we therefore consider network coding applied to only one level of the tree, and develop a framework to identify the optimal level-i at which to place all the recovery effort.

Intuitively, if a given number of coded packets are placed at a level-i low in the tree (i.e. close to the data leaves), recovery is more likely to fail since the number of nodes (and hence potential losses) is higher than at upper levels. However, if recovery at this level is successful, the rewards are high as more data items are likely to have a valid authentication path to this level, and can hence be successfully authenticated since all upper levels can be fully reconstructed. The optimal placement therefore balances the risk against the reward to maximise the probability of verification, as derived next.

4.1. Probability of Verification

We first compute the probability P_{ver} that the receiver can verify an arbitrary received data item D. We denote the packet receipt probability by p (conversely, packet loss probability is $1 - p$), the network coding overhead (i.e. number of recovery packets) by R, and the tree level at which coding is used by i. Our objective is to find the i that maximises P_{ver}. Our model makes the following simplifying assumptions:

- Each data item and each recovery information is transmitted as a separate packet.
- Each packet has independent and identically distributed (iid) probability p of being successfully received (we will show later that this is in some ways a worst-case assumption that gives a lower bound).
- The root of the tree is reliably transmitted (e.g. using an ACK-based protocol or a delayed store-and-transmit mechanism, etc.) and is not subject to loss.
- Network coding for recovery is applied only at a single tree-level in each instance (as argued above).

The probability of verification P_{ver} is computed for an arbitrary packet D depicted in Fig. 5. In reference to the same figure, we define the following:

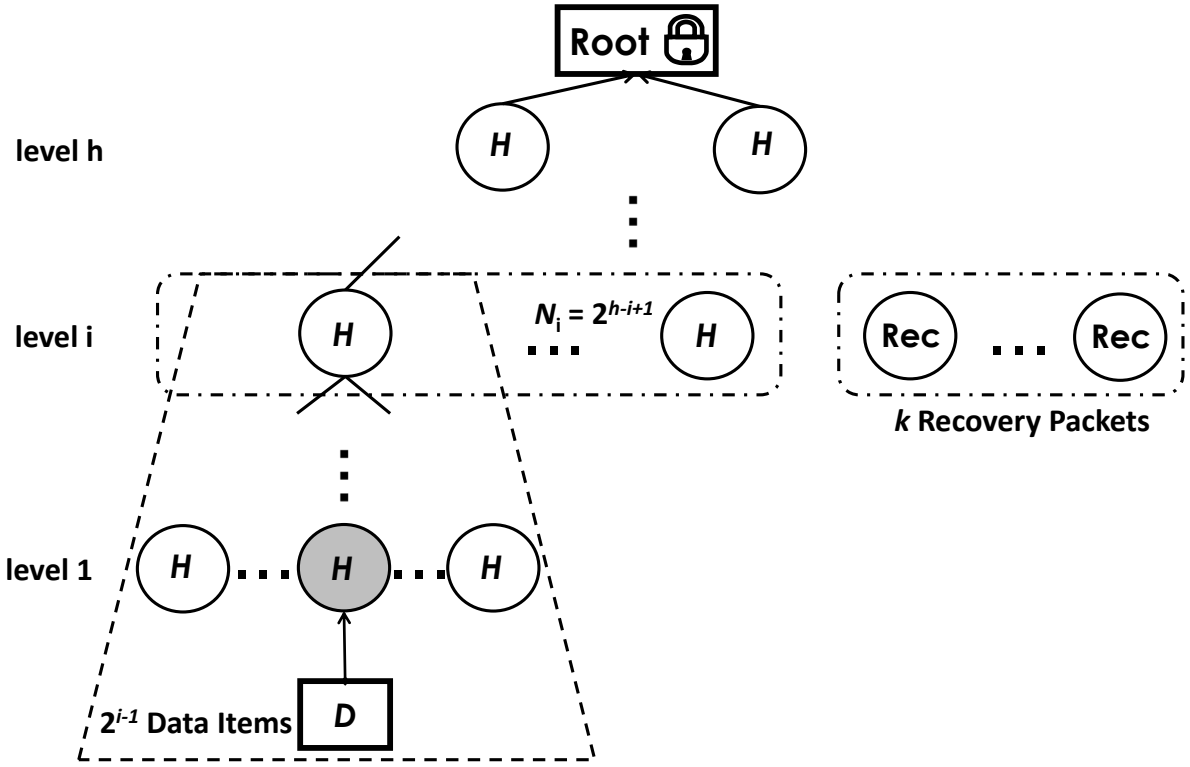

Figure 5: Verifying data item D

Definition. *A node $H_{(i,j)}$ of the tree is said to be* **available** *to the receiver if and only if:*

1. **either reconstruction succeeded:** *for level $i = 1$ (i.e. a leaf node), the child data item D_i was successfully received, or, for $i = 2, 3, \ldots, h$ (i.e. an internal tree node except root), the data items in the subtree rooted at $H_{(i,j)}$, (namely items $D_{(2^i.j)}$ to $D_{(2^i.j-2^i+1)}$) have all been successfully received,*
2. **or recovery succeeded:** *if coding has been applied at level-i, then the sum of the number of nodes reconstructed (from children) at level-i and the number of recovery packets received at least equals the number of nodes $N(i) = 2^{h-i+1}$ at level-i of the tree (this indeed ensures that all nodes at level-i and above are available to the receiver).*

Definition. *In a tree of height h that has $R(i)$ recovery packets at a given level-i , the probability that a received data item D is* ***verifiable*** *by the receiver, is the probability of the intersection of the two events that:*

1. *all other data items in the subtree rooted at level-i, of which D is part, are received, and*
2. *all other nodes at level i are available to the receiver, either by receiving all data items in the corresponding subtrees, or else by recovering any missing level-i nodes using received recovery packets.*

Moreover, the two events in the definition above are independent, by virtue of nodes at the same tree level having disjoint subtrees, and our independent loss model assumption.

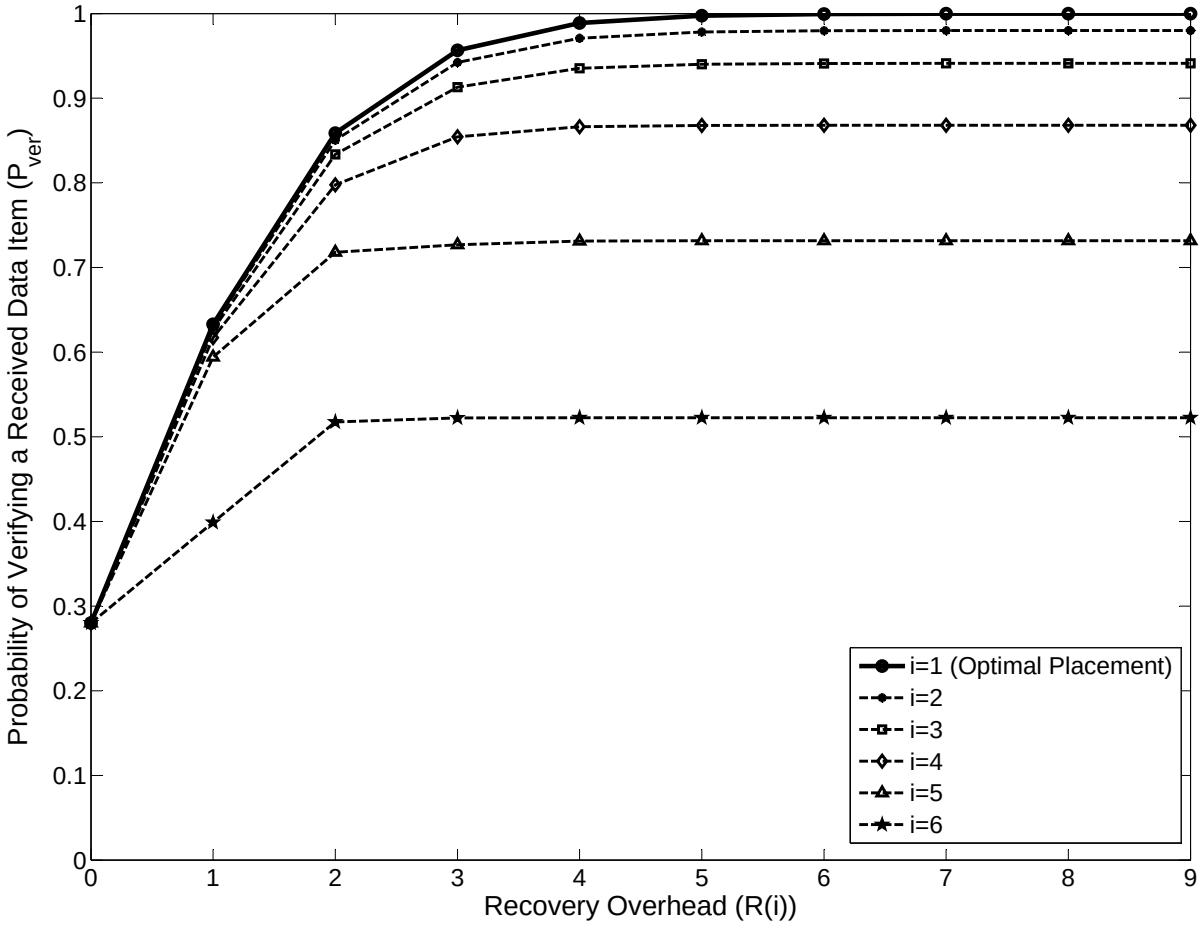

Figure 6: P_{ver} for tree of height $h = 6$ $(1 - p = 2\%)$

Using this, we can compute the probability P_{ver} that a received data item D in the tree is verifiable as:

$$P_{ver} = P\{\text{received all other data items in } D\text{'s subtree}\} \times P\{\text{all other nodes at level } i \text{ are available}\}.$$

Recalling that p denotes probability of successful receipt of a packet, and that the subtree rooted at any node in level-i has 2^{i-1} leaves, we get

$$P\{\text{received all data items in } D\text{'s subtree}\} = p^{2^{i-1}} \equiv \zeta.$$

There are $N(i) = 2^{h-i+1}$ nodes at level i of the tree. To recover D it is imperative that the remaining $N(i) - 1$ nodes at that level be available to the receiver, either by directly reconstructing from received data packets or via recovery. The probability of reconstructing any single node at level i from received data packets is ζ as noted earlier. Then the probability of obtaining exactly l level-i nodes from the $N(i) - 1$ subtrees is

$$P_l^{N(i)-1} = \binom{N(i)-1}{l} \zeta^l (1-\zeta)^{N(i)-l-1}$$

and the probability of receiving exactly k recovery packets from a total of $R(i)$ transmissions is

$$P_k^{R(i)} = \binom{R(i)}{k} p^k (1-p)^{R(i)-k}.$$

Therefore, we take the product of summation of these terms for all possible combinations where the number of reconstructed nodes, l, and received recovery packets, k, such that $l+k \geq N(i)-1$. Finally, we divide by p to condition the probability on successful receipt of the data packet D that is to be verified:

$$P_{ver} = \zeta \sum_{k=0}^{R(i)} \left(P_k^{R(i)} \sum_{l=N(i)-1-k}^{N(i)-1} P_l^{N(i)-1} \right) / p \tag{1}$$

As a validation, when no recovery packets are applied at any layer (i.e. $R(i) = 0$ for all i), the probability of being able to authenticate a received data item simply reduces to the probability of receiving all other data items, i.e. p^{2^h-1}, i.e. the full tree can be reconstructed from just the data items.

4.2. Identifying Optimal i

The above formulation can be used to maximise the probability P_{ver} of verifiability by searching over $i \in [1, h]$. We illustrate the outcome for two plausible scenarios: first, for for non-critical applications (e.g. temperature or heart-rate monitoring), the sensor device aggregates and transmits data items once per minute. To apply a digital signature approximately once per hour, we can construct a tree of height $h = 6$ spanning 64 data items. Our own experiments with bodyworn devices indoors (detailed in the next section) indicate they experience loss in the range $1 \sim 3\%$. Here, we assume packet loss $1-p = 2\%$. Fig.6 depicts how probability of verification P_{ver} improves as more coding packets are transmitted, rising from approximately 30% when no coding is used, to over 99.9% with just 6 recovery packets (an overhead of less than 10%). This trend can be seen more clearly in Fig.7 showing (in log scale) the probability of **not** being able to verify a data item as a function of overhead.

For critical applications (e.g. ECG monitoring), a more appropriate transmission rate is 1 packet/second. If the signature is to be computed once per hour, a tree of height $h = 12$ can be used, spanning 4096 leaves. The number of recovery packets is varied from 0 to 128, and the corresponding verification probability is shown in Fig. 8. Each curve in the figure corresponds to one placement of the recovery packets. The first observation is that when network coding is not used, probability of verification is close to zero, since verification of any data item depends on receipt of all 4096 data items in the tree. As overhead increases, verifiability increases dramatically. The depiction of this on log scale in Fig.7 clearly shows that about 128 recovery packets (corresponding to about 3% overhead) can reduce the inability to verify a data item to nearly 10^{-6}.

The other important observation in the various curves in Fig. 8, each corresponding to a different placement of network coding, is that optimal placement of coding depends on allowed overhead. Indeed, as overhead is increased, moving coding lower in the tree is beneficial. As progressively increasing overhead is applied, it becomes possible to meet the conditions for recovery at lower tree levels. And as noted earlier, if recovery is effected

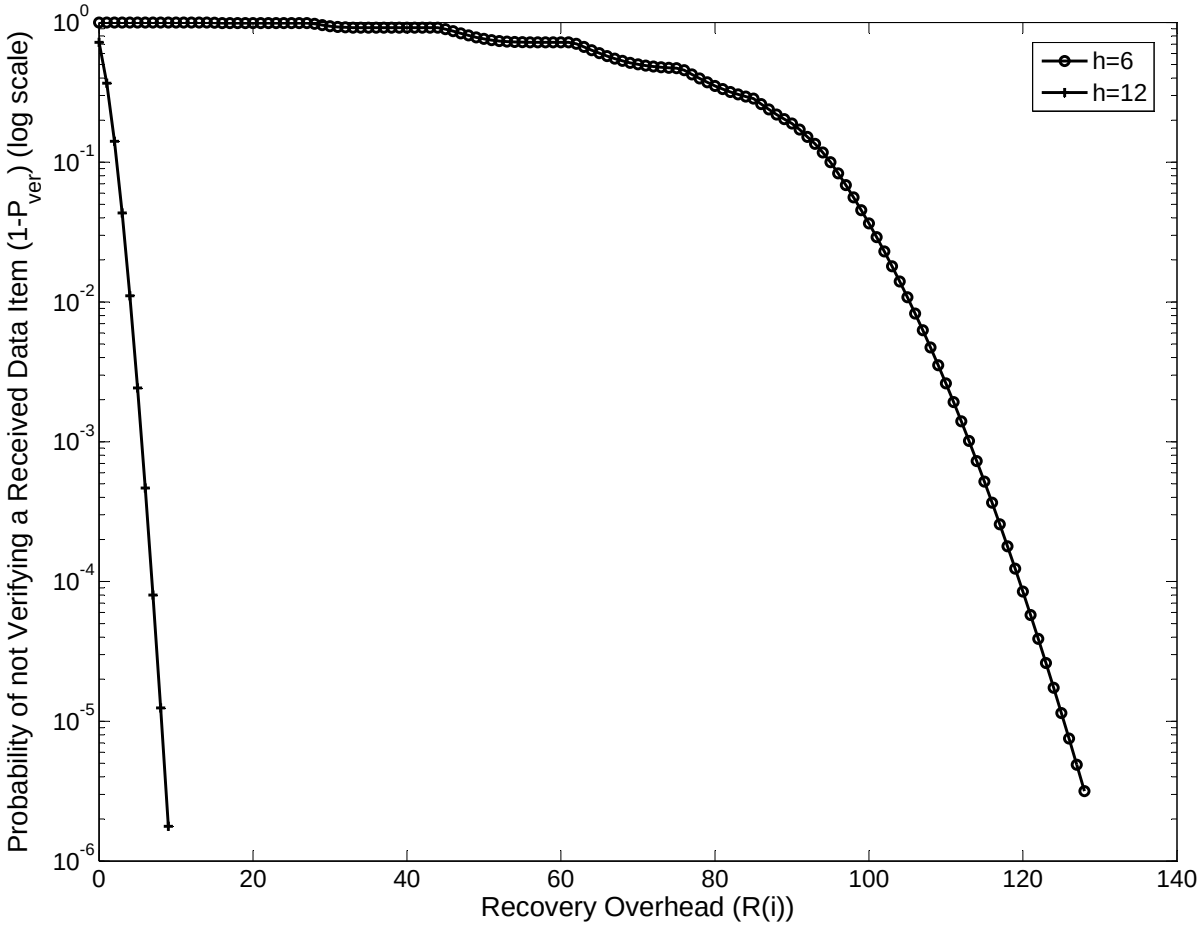

Figure 7: P_{ver} for trees of various heights ($1 - p = 2\%$)

at a lower level, the probability of verification is higher because the authentication path for each data item is correspondingly shorter.

5. Simulation and Experimental Results

We note here a limitation of our optimization framework, namely that it operates on the simplifying assumption of point losses, each packet being independent and identically distributed (iid) with probability p. In practice, losses in wireless sensor networks tend to be bursty [25] and researchers have recently attempted to use Markov modeling to characterize the state of the wireless channel for body area networks [5] [27].

5.1. Gilbert Model

To validate the suitability of our analytical iid-based model, we compare its output with results from a basic 2-state Markov model (as specified by Gilbert [10], and depicted in Fig. 9). In this model there are two states, Receive (or 0), in which a packet is received and Loss (or 1), in which it is dropped, and p_g and q_g are the transition probabilities from state to Receive to Loss and Loss to Receive, respectively. We derive these transition probability values from experimental traces collected using actual bodyworn devices, as we describe next.

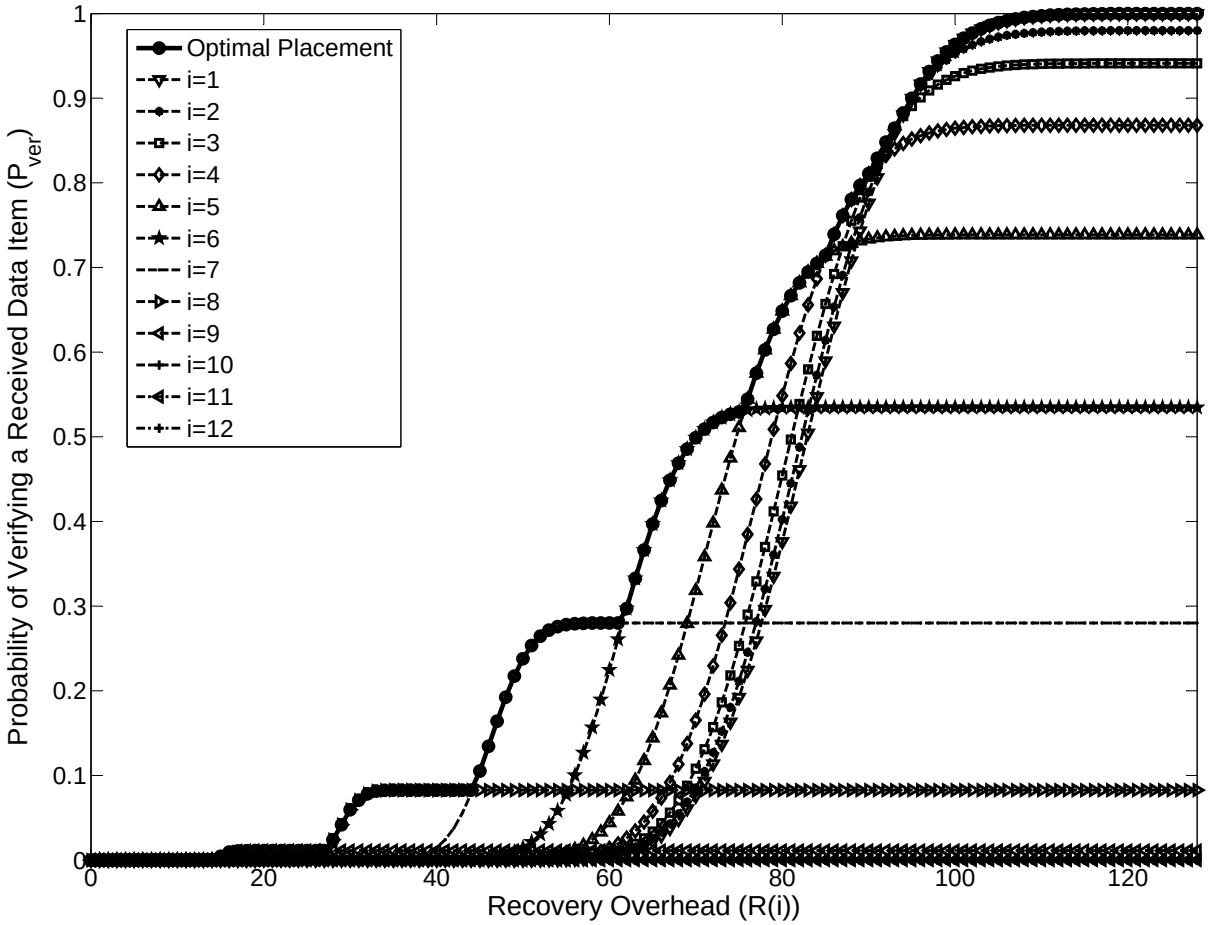

Figure 8: P_{ver} for tree of height $h = 12$ $(1 - p = 2\%)$

5.2. Experimentation

Furthermore, we conduct experiments with a real bodyworn network: a male subject wears two communicating MicaZ motes, one on his right arm, the other at his waist on the left side, and works in an indoor office environment. The wireless channel is sampled at a rate of 1 packet/second at maximum transmission power. We collected traces worth several hours of data, and present here three representative sets as summarized in Table 1: *Low Activity*, in which the subject is sitting and working at his cubicle, *Moderate Activity*, in which the subject occasionally gets up from his desk to walk around the room, and *High Activity*, where the subject ventures outdoors periodically for brief periods. For each scenario, we compute and list in Table 1 the packet loss $(1 - p)$, the average burst length, and the duration of the experiment. The last two columns denote the corresponding state transition probabilities, p_g, the probability the channel transitions from the receiving state to a lossy state, and, q_g the probability that the channel will transition from a lossy to a receive state again.

The packet loss rate is seen to increase with level of activity. The significantly higher loss rate for *High Activity* is also in part due to the outdoor trips (higher loss rate is due to reduced multipath in an outdoor environment). The observed loss characteristics in our scenarios were consistent with other studies of the near-body channel (e.g. [19]),

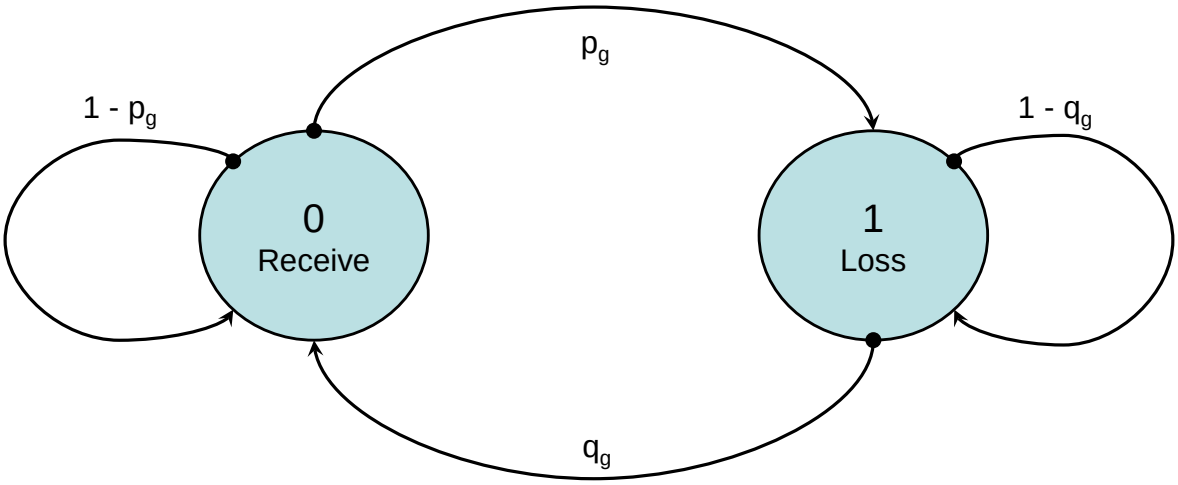

Figure 9: Gilbert Model for generating a 2-state Markov Model for simulating burst losses

Table 1: Specifications of Data Sets collected in Experiments

Data Set	**Pkt Loss (1-p) (%)**	**Avg. Burst Length (s)**	**Duration (minutes)**	p_g (%)	q_g (%)
Low Activity	0.92	1.48	150	0.7	77.5
Moderate Activity	1.28	1.29	300	1.02	77.67
High Activity	2.93	1.67	1.32	1.72	60.68

namely loss $1 - p$ is typically in the range of 1-5%, it is influenced by motion, and it is more pronounced in outdoor environments.

Fig. 10 depicts the variation in burst length for all three activity scenarios and we observe, particularly for *Moderate* and *High Activity* a significant number of losses are of burst length greater than 1.

We input the experimentally derived p_g and q_g parameters into the Gilbert model to simulate packet loss conditions for each of the three activity scenarios for a tree of height $h = 12$.. For each value of applied overhead, $R(i)$, the simulation measures the probability of verification, P_{ver} at every level of the tree and records the best result and the optimal placement, i. Fig. 11 depicts the probability of verification of a received data item for progressively increasing overhead. For comparison, we also plot, for each activity, the results of the analytical iid model, using the corresponding packet loss probability, p.

5.3. Discussion of Results

The very first observation is that simulation results show markedly better performance than the analytical iid model for all three cases. This is especially important for the *High Activity* scenario: for 128 packets of overhead, the analytical iid model can only deliver up to approximately 85% probability of verifiability for a received data item whereas the simulation predicts greater than 95%. Indeed, the verification probability for the simulation results is consistently larger than the analytical iid model (for identical loss rate), indicating that the latter is conservative in its estimate. To understand why, recall that our model assumes iid loss, which results in a more even spread of losses, which is

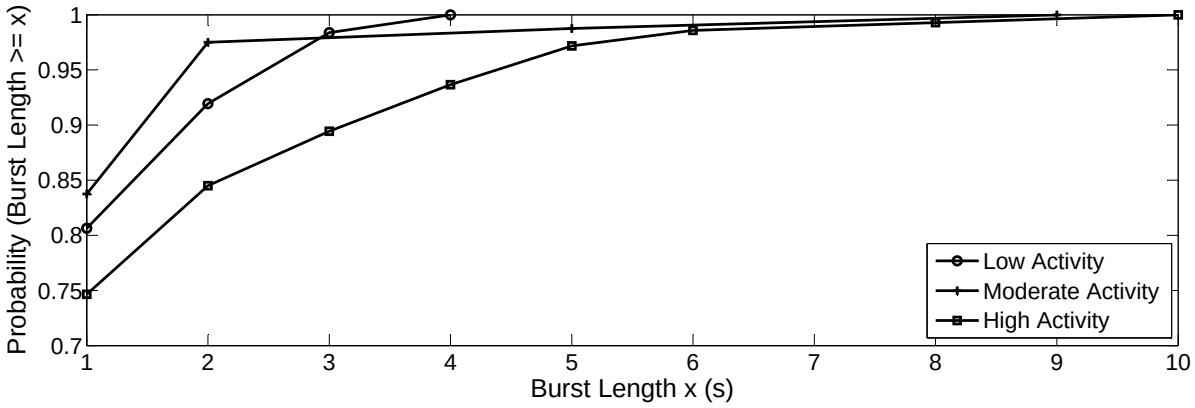

Figure 10: Distribution of burst losses for all three activities

actually worse than having adjacent losses in the tree. For example, referring to the hash tree in Fig. 2, consider the case of a data set where exactly two data items are dropped. If the losses are approximately evenly spaced, say for example D_1 and D_4 in a set of 8 data items (the losses are both in the first half of the set), there will be two missing nodes at level-1 ($H_{(1,1)}$ and $H_{(1,4)}$) and level-2 ($H_{(2,1)}$, $H_{(2,2)}$), and so on up the tree until the losses are contained within a subtree, in this case, level-3 where only $H_{(3,1)}$ is missing. Now if one coded packet is received for level-3, the second half of the data set can be verified, i.e. $D_5 - D_8$, 4 items out of 8. However, if these two losses occurred in a burst, e.g. D_1 and D_2, the effect on the integrity of the tree would be far less. There would be two losses at level-1 ($H_{(1,1)}$ and $H_{(1,2)}$) and one loss each at level-2 (i.e. $H_{(2,1)}$) and level-3 ($H_{(3,1)}$). In this case if a single recovery packet coded for level-2 is received, items $D_3 - D_8$ can be verified, i.e. 6 items out of 8. Bursty losses are therefore less damaging to the integrity of the tree, and less recovery overhead is needed to compensate. Consequently, our analysis using an iid loss model can be treated as a conservative estimate (i.e. lower-bound) of the probability of data item verifiability.

Next, we apply our simulations results to the experimental traces for each of the three activity scenarios. We overlay hash trees of height $h = 12$ on the connectivity traces, and determine, for various coding overheads, the fraction of received data items verifiable at the receiver. The optimal overhead placement, i.e. level i in the tree, is selected as identified by the simulation. The results are shown in Figs. 12-14

The first observation from these plots is that our scheme, when applied to real traffic traces, does indeed corroborate with our analysis to show that an overhead of just 128 extra packets ($\sim 3\%$) can yield effectively near-100% verifiability for data items. This confirms that the scheme has merit in real-life bodyworn scenarios.

And, for each of the three situations, it is encouraging to note that the experimental results are superior to the expected outcome of the simulation results (which we showed earlier performed vastly better than the analytical iid optimization model in Fig. 11). We speculate that this variation is very likely due to using the Gilbert model to model the near-body channel, a simplistic attempt in that it uses only two states, *Receive* and *Loss*. This is suggested by the fact that the difference between simulation and experimental results becomes more pronounced as we move from *Low* to *High Activity* scenarios and the packet loss increases. We can assume that there are other receive and loss states

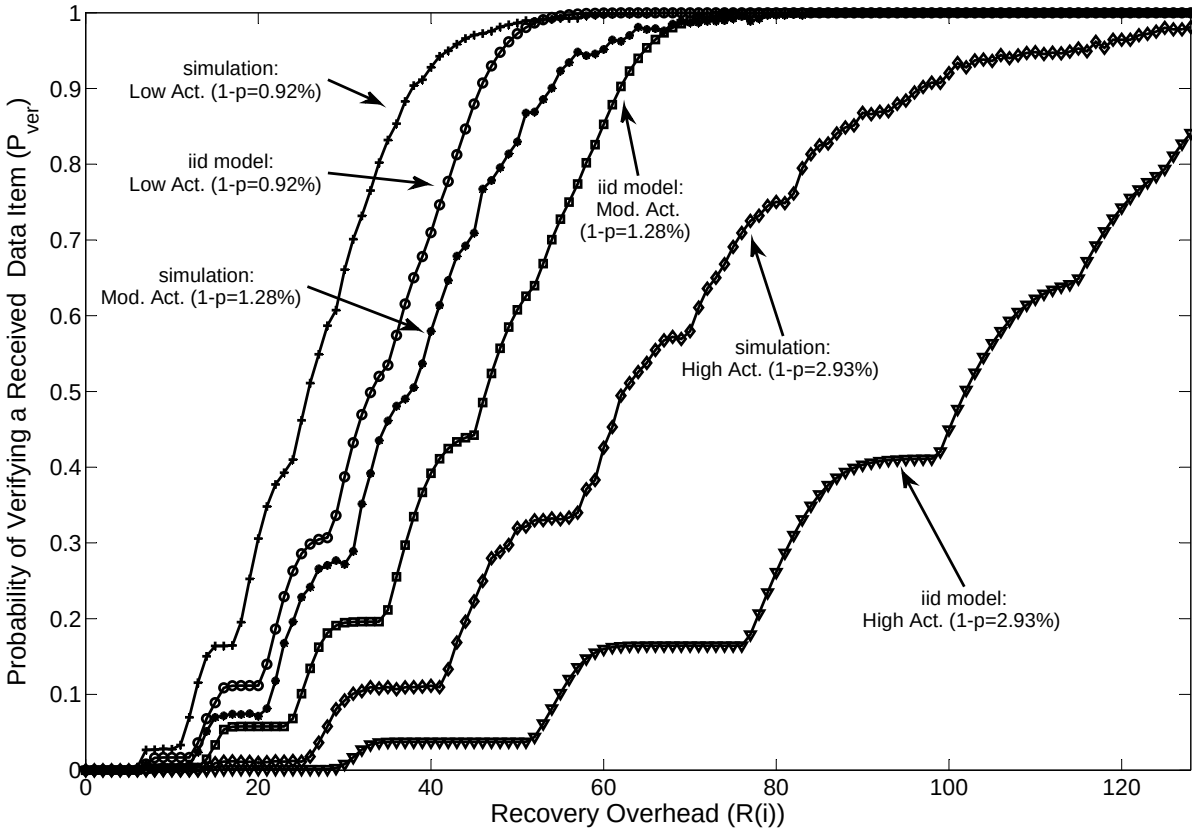

Figure 11: Comparison of results for iid optimization and simulation based on Gilbert model for $h = 12$

involved with varying packet loss probabilities, such as when the subject is moving or is outdoors. Providing a more accurate model for the near-body channel is unfortunately beyond the scope of our current research effort.

In each plot, we also show the predicted probability of verifiability for a loss rate of $1-p = 2\%$ as per the analytical iid model. This is because the scheme has to be optimized prior to deployment, and can thus only use a preset target loss rate (which we pick to be 2%) rather than the actual loss rate in the deployment which may not be accurately known beforehand. The two-state simulation cannot be used to derive optimal values for bootstrap purposes simply because the transition probability values are not known to us beforehand, they depend entirely on device deployment and usage. Therefore, as we demonstrated earlier in this section, the results of the analytical iid model can be considered a very conservative and usable estimate for bootstrapping the scheme.

For low-loss scenarios (in this case $< 2\%$), i.e. *Low* and *Moderate Activity*, the proposed optimization (i.e. $1 - p = 2\%$) compensates more than adequately and yields near perfect authentication. It is observed that even an overhead of approximately 60 recovery packets (i.e. 1.5%) yields more than 95% probability of verification. For *High Activity*, where the loss rate of 2.93% is significantly greater than the preset optimization, an overhead of 128 recovery packets still allows verification of more than 90% of the received data items. If still better performance is desired, the optimization needs to be run for $(1 - p \approx 3\%)$ to locate optimal placement for coding and provision more recovery

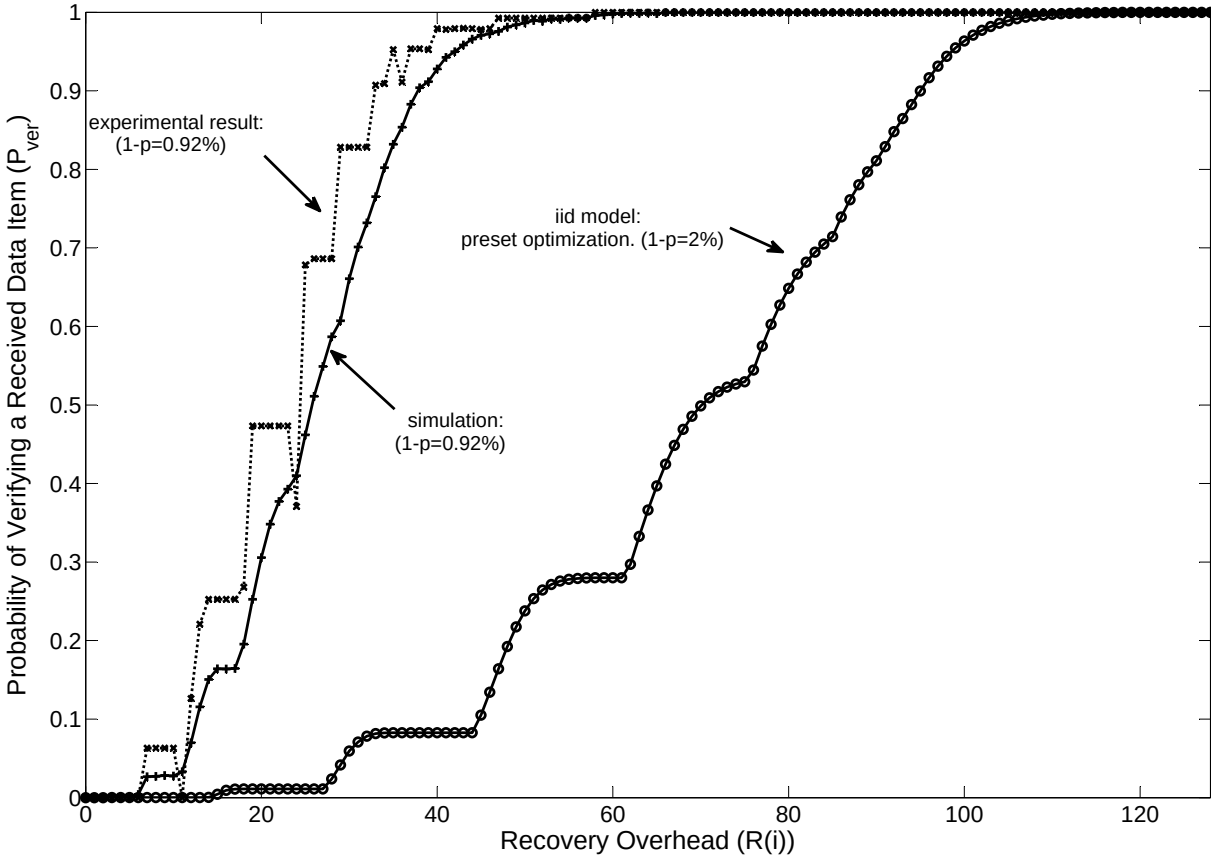

Figure 12: Comparing experimental and simulation results for *Low Activity* ($1 - p = 0.92\%$)

overhead.

The experimental results also show some spikes where the probability of verification drops by very small amounts (approximately 5%) as overhead is progressively increased. This anomaly is due to the fact that we reuse our experimental traces for every situation, and, in some cases, strategic packets get overlaid onto sustained bursty periods in the trace files. We expect these minor fluctuations will be ironed out in large data sets.

Our two-state simulation, which does not seem to be of much value in the bootstrap phase, however, may be used to provide valuable updates to the scheme after deployment. After a period of sustained activity (say, a few hours), the basestation can easily derive the transition probability values using connectivity statistics, as we did earlier, and run the simulation to derive optimal tree placement values and number of recovery packets. These values may differ significantly from the preset values, and the basestation can easily communicate the ideal tree configuration to the sensor device, leading to better performance. Furthermore, this can be an ongoing process, allowing the scheme to adapt to mode of user activity.

6. Conclusion

In this paper, we proposed a low-cost practical solution to authenticate medical data generated by wireless bodyworn sensor devices and stored in the cloud. We employ a Merkle hash tree to amortise digital signature costs and leverage network coding to make

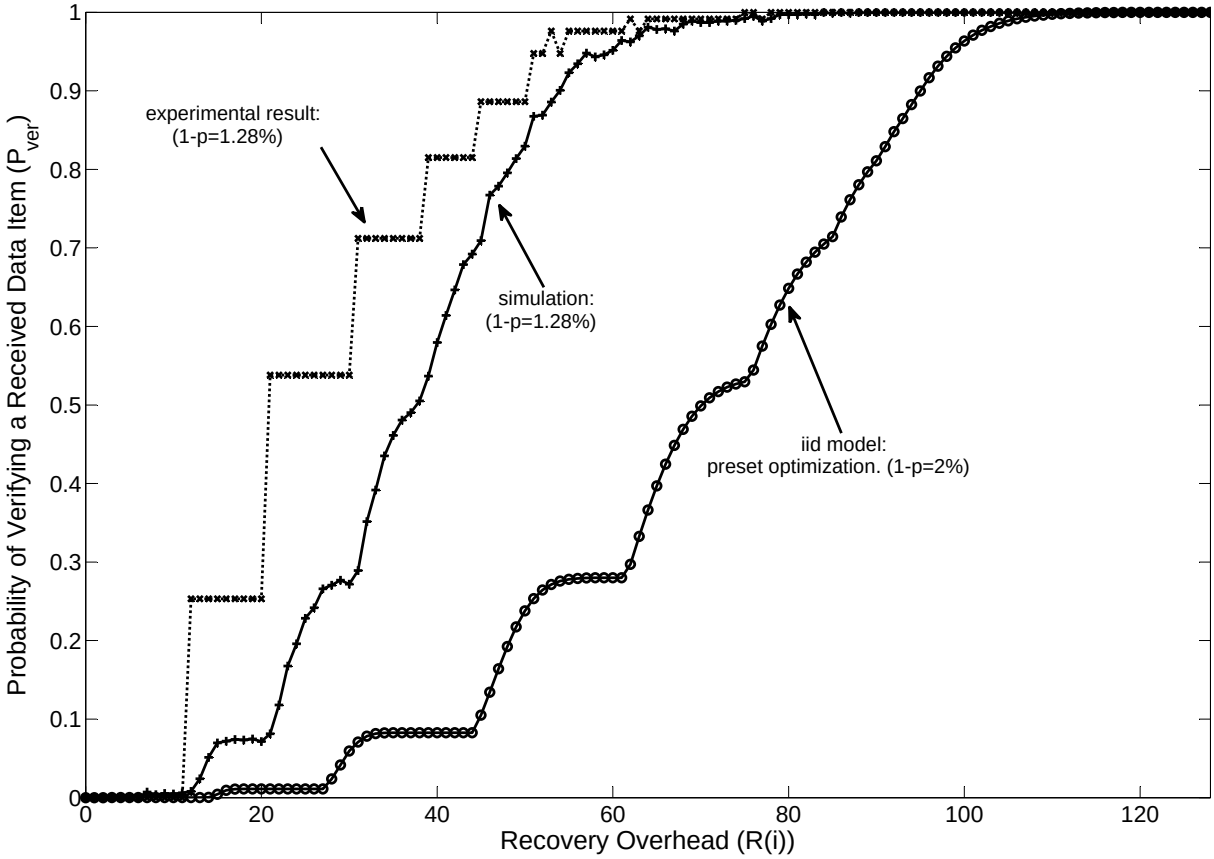

Figure 13: Comparing experimental and simulation results for *Moderate Activity* $(1 - p = 1.28\%)$

the authentication scheme robust to packet loss. We provide an optimisation framework so that network coding is best used to maximise data verification probability for a given loss environment and constraint on overhead. Furthermore, we specify a simulation model which attempts to bridge the gap between our analytical iid model and real-life packet loss conditions. We validate our findings with experimental data collected using real bodyworn devices. Our results indicate that, in a typical indoor environment, over 99% of the data can be authenticated with as low as $3 - 5\%$ overheads in transmission. We believe our proposed solution for ensuring ironclad authenticity of medical data is a positive step towards integrating bodyworn sensors into current cloud-based healthcare systems.

For future work, we plan to implement and trial our solution and enhance it to provide further security functionality, such as data provenance and confidentiality, and coordinate data generated by multiple bodyworn sensors.

References

[1] A. Pannetrat, R. M., 2003. Efficient Multicast Packet Authentication. In: NDSS. San Diego.

[2] Apple Inc., Retrieved 19 November, 2010. Sensor Strip. `http://www.patentlyapple.com/patently-apple/2010/03/body-area-networks-apple-sensor-strips-the-iphone.html`.

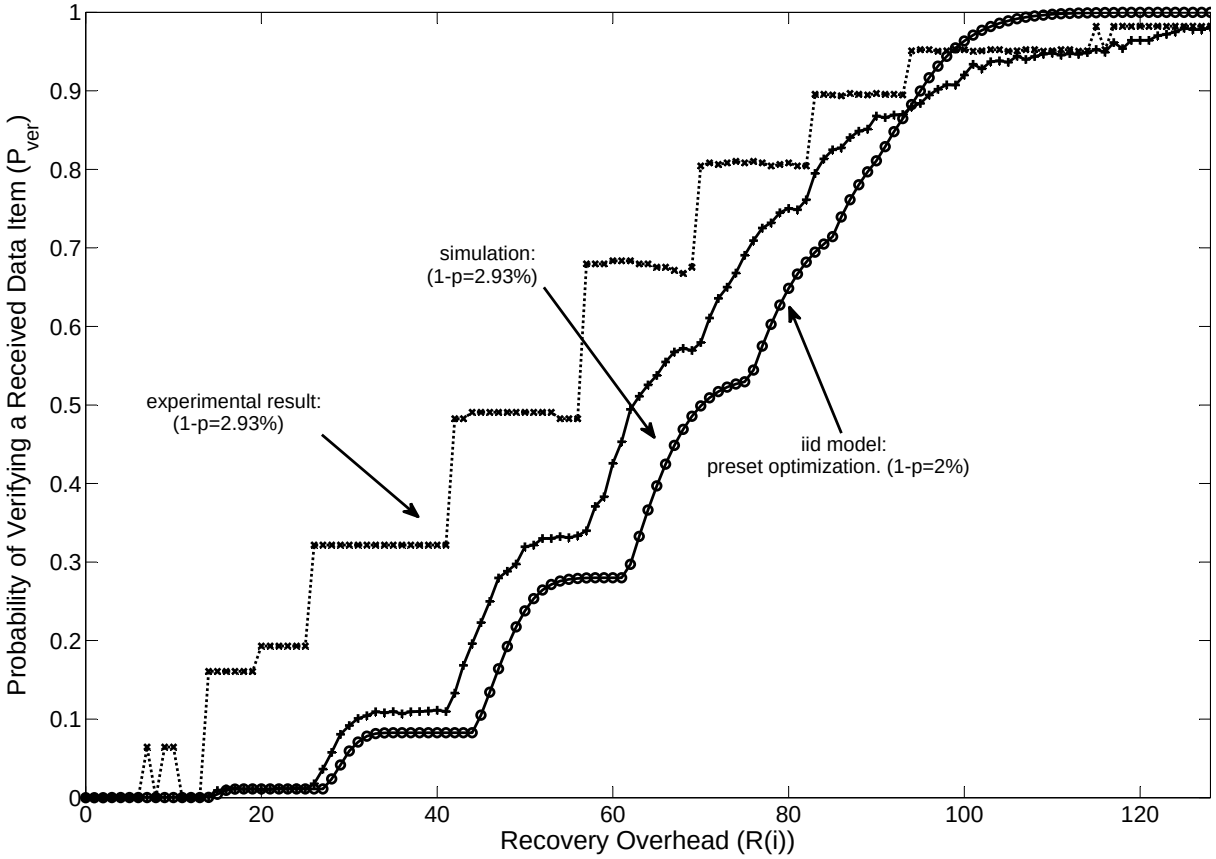

Figure 14: Comparing experimental and simulation results for *High Activity* ($1 - p = 2.93\%$)

[3] c. Buratti, D'Errico, R., Maman, M., Martelli, F., Rosini, R., Verdone, R., 2011. Design of a Body Area Network for Medical Applications: the WiserBAN Project. In: Proceedings of the 4th ACM International Symposium on Applied Sciences in Biomedical and Communication Technologies. p. 164.

[4] CEA-Leti Institute, 2 June, 2013. CORMORAN Project Exploring Ways to Improve Cooperation In and Between Wireless Body Area Networks. `http://goo.gl/cfPMDN`.

[5] Chaganti, V. G., June 2011. Semi-Markov Modeling for Body Area Networks. In: International Conference on Communications (ICC). Kyoto.

[6] COLLAGE: Collaboration on Ageing, Retrieved 13 Aug., 2013. Body Area Networks for Continuous Monitoring of Critical Parameters. `http://www.collage-ireland.eu/wp-content/uploads/2013/04/BAN-PDF.pdf`.

[7] Deng, J., Han, R., Mishra, S., 2006. Secure Code Distribution in Dynamically Programmable Wireless Sensor Networks. In: 5th International conference on Information Processing in Sensor Networks. Nashville, Tennessee, USA, pp. 292–300.

[8] EPSRC funded project (EP/D076943), Retrieved April, 2010. SESAME: SEnsing for Sport And Managed Exercise. http://www.sesame.ucl.ac.uk/index.html.

[9] Gennaro, R., Rohatgi, P., 1997. How to Sign Digital Streams. In: CRYPTO'97. pp. 180–197.

[10] Gilbert, E. N., 1960. Capacity of a Burst-Noise Channel. Bell System Technical Journal 39.

[11] Golle, P., Modadugu, N., 2001. Authenticating Streamed Data in the Presence of Random Packet Loss. In: ISOC Network and Distributed System Security Symposium. pp. 13–22.

[12] Graham-Rowe, D., October 2010. Body Organs can Send Status Updates to Your Cellphone. New Scientist.

[13] Habib, A., Xu, D., Atallah, M., Bhargava, B., Chuang, J., 2005. A Tree-Based Forward Digest Protocol to Verify Data Integrity in Distributed Media Streaming. IEEE Transactions on Knowledge and Data Engineering 17 (7), 1010–1014.

[14] Hasan, M. A., July 2000. Look-up Table-Based Large Finite Field Multiplication in Memory Constrained Cryptosystems. IEEE Transactions on Computers 49.

[15] Hyun, S., Ning, P., Liu, A., Du, W., April 2008. Seluge: Secure and DoS-Resistant Code Dissemination in Wireless Sensor Networks. In: ACM/IEEE IPSN. St. Louis.

[16] Konstantas, D., Jones, V., Herzog, R., 2002. MobiHealth - innovative 2.5 / 3G Mobile Services and Applications for Healthcare. In: IST Mobile and Wireless Telecommunications Summit. Thessaloniki, Greece.

[17] Lun, D. S., Medard, M., Effros, M., 2004. On Coding for Reliable Communication over Packet Networks. In: 42nd Annual Allerton Conference on Communication, Control and Computing.

[18] Merkle, R. C., 1987. A Digital Signature Based on a Conventional Encryption Function. In: Advances in Cryptology - CRYPTO'87. pp. 369–378.

[19] Natarajan, A., de Silva, B., Yap, K.-K., Motani, M., September 2009. Link Layer Behavior of Body Area Networks at 2.4 GHz. In: MobiCom. ACM, Beijing.

[20] Park, J. M., Chong, E. K. P., Siegel, H. J., 2003. Efficient Multicast Stream Authentication Using Erasure Codes. ACM Transactions on Information and System Security 6.

[21] Perrig, A., Canetti, R., Tygar, J. D., Song, D., May 2000. Efficient Authentication and Signing of Multicast Streams over Lossy Channels. IEEE Symposium on Security and Privacy, 56–73.

[22] Piotrowski, K., Langendoerfer, P., Peter, S., 2006. How Public Key Cryptography Influences Wireless Sensor Node Lifetime. In: Proceedings of the Fourth ACM Workshop on Security of Ad Hoc and Sensor Networks. Alexandria, Virginia, USA, pp. 169–176.

[23] Sanders, P., Egner, S., Tolhuizen, L., 2003. Polynomial Time Algorithms for Network Information Flow. In: 15th ACM Symposium on Parallel Algorithms and Architectures.

[24] Service, A. R., 2009. Market for Wearable Wireless Sensors to Grow to More than 400 Million Devices by 2014. online.

[25] Srinivasan, K., Kazandjieva, M., Agarwal, S., Levis, P., Nov. 2008. The $\beta - factor$: Measuring Wireless Link Burstiness. In: SenSys. Raleigh.

[26] Toumaz Technology Ltd., Retrieved 19 November, 2010. Sensium Life Platform. `http://www.toumaz.com/page.php?page=sensium_intro`.

[27] Tselishchev, Y., June 2011. Reducing Transmission Losses in Body Area Networks using Variable TDMA Scheduling. In: WOWMOM. Tuscany.

[28] University of Bologna, Retrieved 13 Aug., 2013. CuPiD: Closed-loop System for Personalized and at-home Rehabilitation of People with Parkinson's Disease. `http://www.cupid-project.eu/`.

[29] Wong, C. K., Lam, S. S., August 1999. Digital Signatures for Flows and Multicasts. IEEE/ACM Transactions on Networking 7 (4).

[30] Xiao, S., Dhamdhere, A., Sivaraman, V., Burdett, A., January 2009. Transmission Power Control in Body Area Sensor Networks for Healthcare Monitoring. IEEE Journal on Selected Areas in Communications (JSAC) 27 (1), 37–48, special Issue on Body Area Networking.

[31] Zhang, Z., Sun, Q., Wong, W.-C., July 2005. A Proposal of Butterfly-graph Based Stream Authentication over Lossy Networks. In: ICME. Amsterdam.