Authentication of Lossy Data in Body-Sensor Networks for Healthcare Monitoring

Syed Taha Ali
University of New South Wales
Sydney, Australia
Email: taha@student.unsw.edu.au

Vijay Sivaraman
University of New South Wales
Sydney, Australia
Email: vijay@unsw.edu.au

Diethelm Ostry
CSIRO ICT Centre
Sydney, Australia
Email: diet.ostry@csiro.au

***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 mobile phone, and then archived over the Internet at a central database. Authenticating the data 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 develop 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 ingredient in the integration of wearable medical monitoring devices into current healthcare systems.**

I. 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 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, enabling continuous monitoring (and possible treatment) of patients in their homes. Wearable platforms have recently begun to appear for personalized healthcare: the Sensium Digital Plaster [1] 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 [3]. ABI research predicts 59 million wearable home health devices will be in use by 2014 [4].

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 should be traceable back to the originating device, it should be non-repudiable, and should not be forgeable by anyone, including the patient himself.

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 mobile phone. 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) 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. These requirements necessitate data authenticity be guaranteed by the source, namely the bodyworn device, rather than an intermediate transit point.

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).

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 can be made publicly accessible). However, digital signatures are computationally expensive, typically two to three orders of magnitude more costly than symmetric-key operations.

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-area network, for which extensive experimentation has indicated that movement and changes in body orientation can induce significant loss [5], [6]), but should instead be designed to be robust to data loss. Existing data stream authentication mechanisms in the literature are not easily adaptable to online scenarios involving devices with very low computation, memory, communication, and power resources.

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 the best use of coding to maximize the fraction of data items that can be successfully verified. **Third**, we validate our scheme 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 healthcare monitoring.

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

II. 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.

A. System Architecture

Recently proposed deployments of medical sensor networks are geared towards interconnecting and integrating various devices such as wireless sensors, personal digital assistants (PDAs), mobile phones, tablets, PC-class systems, online databases, etc. Examples include the MobiHealth [7] and UbiMon [8] projects targeting continuous patient monitoring and value-added healthcare.

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 device to read his blood pressure, ECG, etc. once per second. These readings are periodically communicated to a basestation i.e. a mobile phone or PDA, that may incorporate a utility to view the physiological readings, and uploads the data stream over the Internet to a centralized database. 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, investigate malpractice, and assign liabilities. Our aim is to secure this data from tampering

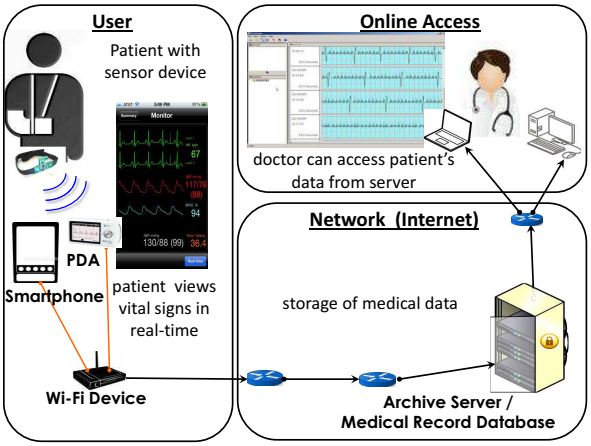


Figure 1. System Architecture

throughout the network while allowing it to be easily viewed and authenticated.

B. 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 [5] for bodyworn devices, however, it is also generally low [6] 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.

Our focus 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 logging on to the database. We do not consider more advanced attacks in this paper, such as denial-of-service or jamming attacks.

C. Related Work

In this section, 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.

1) 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 [9]. 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. Authentication schemes therefore typically utilize hash functions to amortize signature costs over large sets of data, which we examine next.

2) Hash Chaining: Hash chaining [10] 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. [11], [12] and [13] 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 [14] 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 [15] 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.

3) Hash trees: Hash trees, first proposed by Ralph Merkle [16], 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 the cost of the signature. 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. Alternatively, the receiver can authenticate an individual data item in the

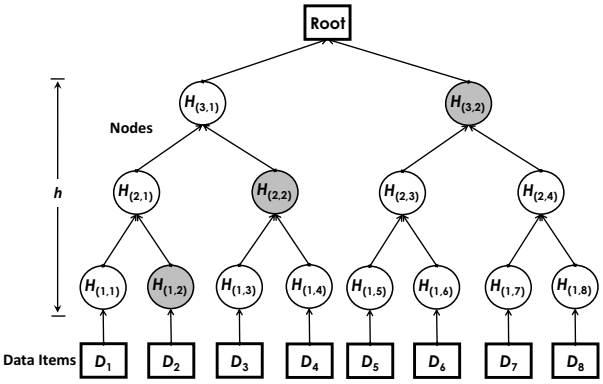


Figure 2. An 8-leaf Hash Tree

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 [17] , 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 [18] 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.

[19] and [20] 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.

4) 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 [14] 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 [14], [18], and [17] 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 potentially comprising 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 data streams for the unique case of severely resource-constrained devices in a lossy environment.

D. 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 [21], 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 [22]. Next we show how network coding can be applied to hash trees to make authentication robust to loss.

III. Our Scheme for Authenticating Lossy Data

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

A. 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.

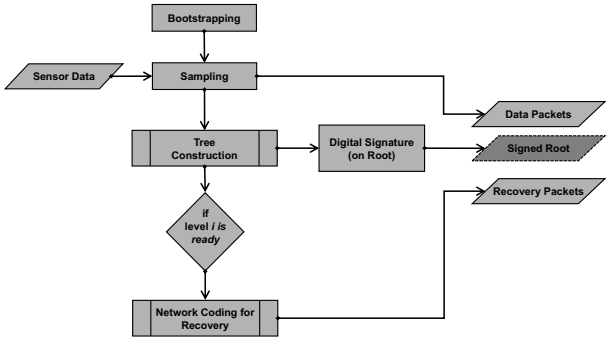


Figure 3. Process flow at sender

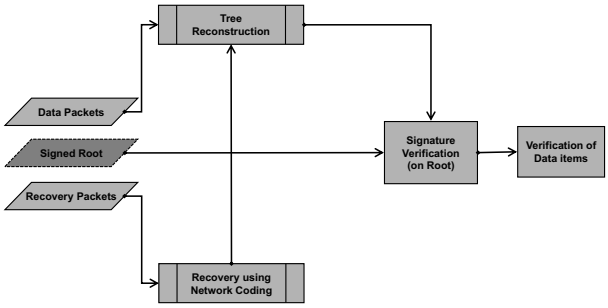


Figure 4. Process flow at receiver

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

B. 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. For example, bodyworn sensors transmitting

a 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, to perform the signature operation approximately once per hour. 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.

IV. 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.

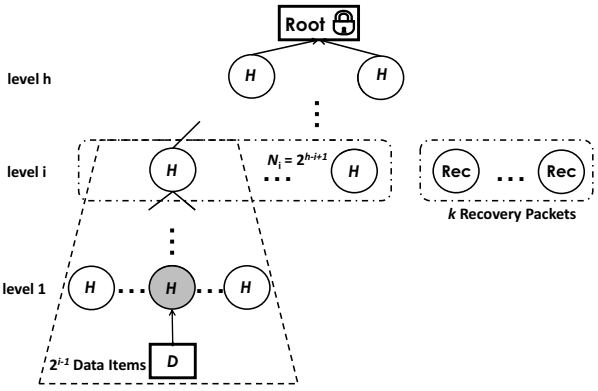


Figure 5. Verifying data item *D*

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.

A. 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 transmitted using a reliable ACK-based mechanism and is not subject to loss.
- Network coding for recovery is applied only at a single tree-level in each instance (as argued above).

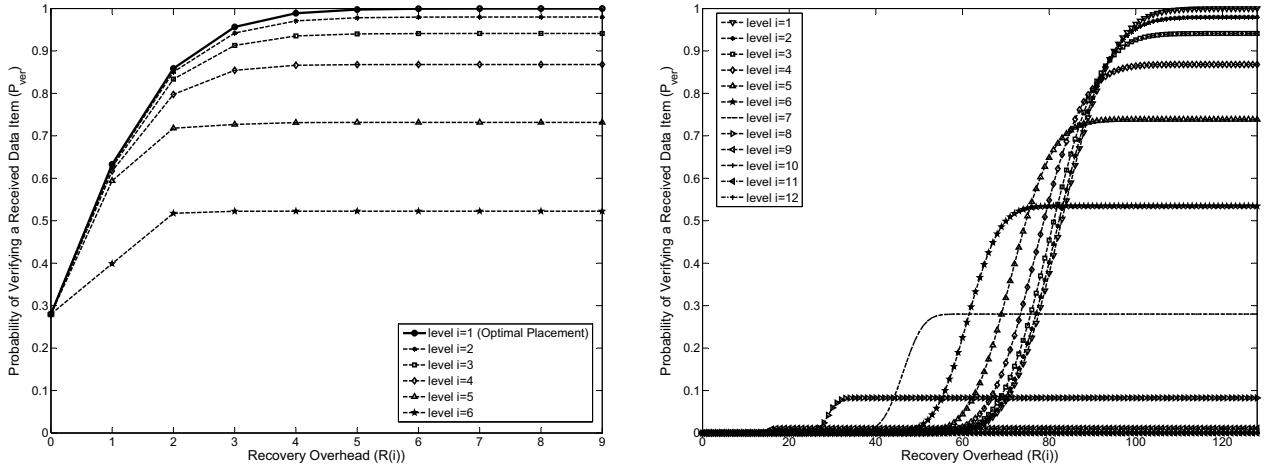
The probability of verification P_{ver} is computed for an arbitrary packet D depicted in Fig. 5. In reference to the same figure, we define the following:

Definition 1: 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 2: 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



(a) tree of height $h = 6$

(b) tree of height $h = 12$

Figure 6. P_{ver} for trees of various heights ($1 - p = 2\%$)

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. 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:

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

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.

B. 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(a) 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. 6(b). Each curve in the figure corresponds to one placement of the recovery packets. The first observation is that when network

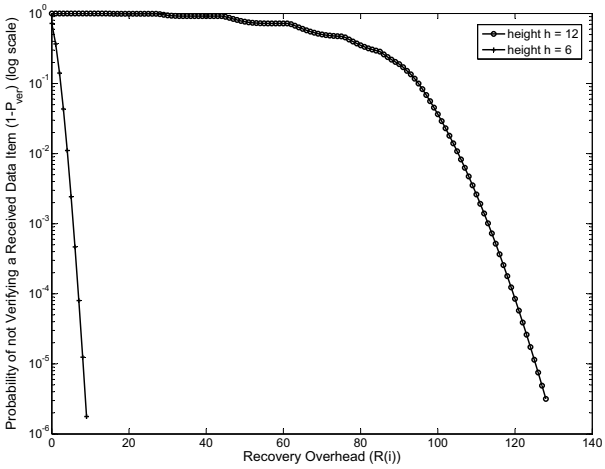


Figure 7. P_{ver} for trees of various heights ($1 - p = 2\%$)

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

V. Experimental Results

We compare analytical results with experimentation 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 several traces worth several hours of data, and present here two representative sets as summarized in Table I: *Low Activity*, in which the subject works at his cubicle and occasionally walks about the office, and *High Activity*, where the subject ventures outdoors periodically for brief periods (higher loss rate is due to reduced multipath). The observed loss characteristics in our scenarios were consistent with other studies of the near-body channel (e.g. [6]), namely loss $1-p$ is typically in the range of 1-5%, it is influenced by motion, it is more pronounced in outdoor environments, and packet loss run lengths are mostly of size 1.

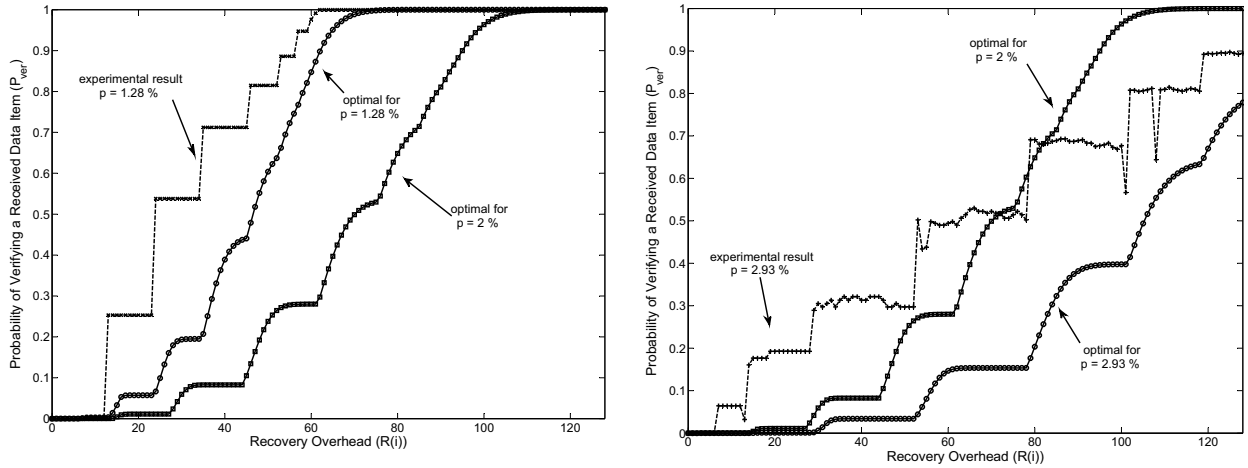
We overlay hash trees of height $h = 12$ over these connectivity traces, and determine, for various coding overheads, the fraction of received data items verifiable at the receiver. The results are shown in Fig. 8, which also plots the predicted probability of verification P_{ver} from our analytical model. The first observation from these plots is that our scheme, when applied to real traffic traces, does corroborate with analysis to show that an overhead of just a 5-10% can yield effectively near-100% verifiability for data items. This confirms that the scheme has merit in real-life scenarios that may exhibit bursty losses.

Table I
Specifications of Data Sets collected in Experiments

Data Set	Packet Loss (1-*p*) (%)	Average Loss Run Length (s)	Duration (minutes)
Low Activity	1.28	1.29	132
High Activity	2.93	1.67	300

Indeed, the verification probability in the experimental trace is consistently larger than from our analytical 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 actually turns out to be worse than having adjacent losses in the tree. For example, referring to 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. 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.

In Fig. 8 we also show in each plot the predicted probability of verifiability for a loss rate of $1 - p = 2\%$. This is because the scheme has to be optimized prior to deployment, and can thus only use a 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. For *Low Activity*, the proposed optimization (i.e. $1 - p = 2\%$) compensates adequately and yields near perfect authentication but is not as effective for *High Activity* due to the higher loss rate in the environment. For this situation, the optimization can authenticate up to 90% of the received data items. The experimental results show some spikes due to sustained burst lengths in



(a) for *Low Activity* ($1 - p = 1.28\%$)

(b) for *High Activity* ($1 - p = 2.93\%$)

Figure 8. Comparing experimental and optimization results ($h = 12$)

the traces (mostly when subject is venturing outdoors) which cause verification probability to fluctuate unpredictably. 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 overhead.

VI. Conclusion

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

References

[1] Toumaz Technology Ltd., *Sensium Life Platform*, http://www.toumaz.com, retrieved 19 November, 2010.

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

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

[4] ABI Research Service, *Market for Wearable Wireless Sensors to Grow to More than 400 Million Devices by 2014*, http://www.abiresearch.com, 2009.

[5] S. Xiao, A. Dhamdhere, V. Sivaraman, and A. Burdett, "Transmission Power Control in Body Area Sensor Networks for Healthcare Monitoring," *JSAC*, vol. 27, no. 1, 2009.

[6] A. Natarajan, B. de Silva, K.-K. Yap, and M. Motani, "Link Layer Behavior of Body Area Networks at 2.4 GHz," in *MobiCom'09*. Beijing: ACM.

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

[8] J. W. Ng, B. P. Lo, O. Wells, M. Sloman, N. Peters, A. Darzi, C. Toumazou, and G.-Z. Yang, "Ubiquitous Monitoring Environment for Wearable and Implantable Sensors (UbiMon)," in *UbiComp*, Nottingham, UK, September 2004.

[9] K. Piotrowski, P. Langendoerfer, and S. Peter, "How Public Key Cryptography Influences Wireless Sensor Node Lifetime," in *ACM SASN'06*, Alexandria, USA, 2006, pp. 169–176.

[10] R. Gennaro and P. Rohatgi, "How to Sign Digital Streams," in *CRYPTO'97*, 1997, pp. 180–197.

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

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

[13] Z. Zhang, Q. Sun, and W.-C. Wong, "A Proposal of Butterfly-graph Based Stream Authentication over Lossy Networks," in *ICME*, Amsterdam, July 2005.

[14] J. M. Park, E. K. P. Chong, and H. J. Siegel, "Efficient Multicast Stream Authentication Using Erasure Codes," *ACM TRANSACTIONS ON INFORMATION AND SYSTEM SECURITY*, vol. 6, 2003.

[15] R. M. A. Pannetrat, "Efficient Multicast Packet Authentication," in *NDSS*, San Diego, 2003.

[16] R. C. Merkle, "A Digital Signature Based on a Conventional Encryption Function," in *CRYPTO'87*, pp. 369–378.

[17] C. K. Wong and S. S. Lam, "Digital Signatures for Flows and Multicasts," *IEEE/ACM Transactions on Networking*, vol. 7, no. 4, Aug. 1999.

[18] A. Habib, D. Xu, M. Atallah, B. Bhargava, and J. Chuang, "A Tree-Based Forward Digest Protocol to Verify Data Integrity in Distributed Media Streaming," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 7, pp. 1010–1014, 2005.

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

[20] J. Deng, R. Han, and S. Mishra, "Secure Code Distribution in Dynamically Programmable Wireless Sensor Networks," in *ACM/IEEE IPSN*, Tennessee, USA, 2006, pp. 292–300.

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

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

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