Secure Ad Hoc Trust Initialization and Key Management in Wireless Body Area Networks

MING LI, Utah State University
SHUCHENG YU, University of Arkansas at Little Rock
JOSHUA. D. GUTTMAN, Worcester Polytechnic Institute
WENJING LOU, Virginia Tech
KUI REN, Illinois Institute of Technology

The body area network (BAN) is a key enabling technology in e-healthcare. An important security issue is to establish initial trust relationships among the BAN devices before they are actually deployed and generate necessary shared secret keys to protect the subsequent wireless communications. Due to the ad hoc nature of the BAN and the extreme resource constraints of sensor devices, providing secure as well as efficient and user-friendly trust initialization is a challenging task. Traditional solutions for wireless sensor networks mostly depend on key predistribution, which is unsuitable for a BAN in many ways. In this article, we propose *group device pairing* (GDP), a user-aided multi-party authenticated key agreement protocol. Through GDP, a group of sensor devices that have no pre-shared secrets establish initial trust by generating various shared secret keys out of an unauthenticated channel. Devices authenticate themselves to each other with the aid of a human user who performs visual verifications. The GDP supports fast batch deployment, addition and revocation of sensor devices, does not rely on any additional hardware device, and is mostly based on symmetric key cryptography. We formally prove the security of the proposed protocols, and we implement GDP on a sensor network testbed and report performance evaluation results.

Categories and Subject Descriptors: C.2.0 [**Computer–Communication Networks**]: General—*Security and protection*; C.2.1 [**Computer–Communication Networks**]: Network Architecture and Design—*Wireless communication, Network topology*; C.4 [**Computing Systems Organization**]: Performance of Systems; D.4.6 [**Operating Systems**]: Security and Protection—*Cryptographic controls*; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection

General Terms: Security, Design, Experimentation

Additional Key Words and Phrases: Trust establishment, key management, usable security, device pairing, body area networks, efficiency

A preliminary version of this paper [Li et al. 2010] appeared in *Proceedings of the 29th Conference of on Computer Communications (InfoCom'10)*.
This work was supported in part by the U.S. National Science Foundation under grants CNS-0716306, CNS-0831628, CNS-0746977, and CNS-0831963.
Authors' addresses: M. Li, Department of Computer Science, Utah State University, 4205 Old Main Hill, Logan, UT 84322; email: ming.li@usu.edu; S. Yu, Department of Computer Science, University of Arkansas at Little Rock, 2801 S. University Ave, Little Rock, AR 72204; email: sxyu1@ualr.edu; J. D. Guttman, Department of Computer Science, Worcester Polytechnic Institute, 100 Institute Road, Worcester, MA 01609; email: guttman@wpi.edu; W. Lou, Department of Computer Science, Virginia Tech, 7054 Haycock Road, Falls Church, VA, 24061; email: wjlou@vt.edu; K. Ren, Department of Electrical and Computer Engineering, Illinois Institute of Technology, 3301 Dearborn St, Siegel Hall 319, Chicago, Illinois 60616; email: kren@iit.edu.

DOI: http://dx.doi.org/10.1145/2422966.2422975

ACM Reference Format:
Li, M., Yu, S., Guttman, J. D., Lou, W., and Ren, K. 2013. Secure ad hoc trust initialization and key management in wireless body area networks. ACM Trans. Sensor Netw. 9, 2, Article 18 (March 2013), 35 pages.
DOI: http://dx.doi.org/10.1145/2422966.2422975

1. INTRODUCTION

In recent years, the interoperable medical device (IMD) [Venkatasubramanian et al. 2010] has emerged as an enabling technique for modern e-healthcare systems, which would revolutionize hospital treatment [Lorincz et al. 2004; Hanson et al. 2009; Jovanov et al. 2005; Li et al. 2010a]. Traditional medical devices usually operate separately, while IMDs are able to interoperate with each other—they are small wearable or implantable medical devices that are capable of sensing, storing, processing, and transmitting data via wireless communications. IMDs afford many advantages to the patient including improved safety, more accurate diagnosis, and better context awareness for caregivers [Venkatasubramanian et al. 2010].

A network of IMDs is often referred to as a wireless body area network (BAN). It may consist of multiple IMDs of different types—they could be placed in, on, or around a patient's body while fulfilling the common goal of patient monitoring. In addition, a controller (a hand-held device like a PDA or smart phone) is usually associated with each patient which collects, processes, and transmits the sensor data to the upper tier of the network for healthcare records. A typical structure of the BAN and its relationship with the e-healthcare system is depicted in Figure 1.

The BAN is designed to satisfy a wide range of applications, such as ubiquitous health monitoring (UHM) [Jovanov et al. 2005] and emergency medical services (EMS) [Lorincz et al. 2004]. The UHM features long-term and consistent monitoring of a patient's health status and surrounding environment, while the EMS requires real-time medical data collection and reporting.

Unlike conventional sensor networks, a BAN deals with medical information, which has stringent requirements for security and privacy. It is critical to protect this information from eavesdropping, malicious modification, and unauthorized access, etc. Trust among the BAN devices is crucial for realizing these security requirements, especially regarding authenticated shared (symmetric) secret keys that enable cryptographic functions, such as encryption and integrity check. However, in traditional wireless sensor networks (WSNs), the secret keys are usually predistributed before network deployment. The existing methods for key distribution in WSNs can be divided into several categories. (1) Rely on knowledge of the network topology [Perrig et al. 2002]; (2) require less topology information but need the sensors to store a large number of keys [Eschenauer and Gligor 2002; Chan et al. 2003; Di Pietro et al. 2003; Du et al. 2005; Liu and Ning 2003; Liu et al. 2008]; (3) assume the existence of root of trust from certain central entities [Zhu et al. 2003, 2006] or rely on public key infrastructure (PKI) [Malan et al. 2004].

However, key predistribution is not suitable for a BAN in several ways. First, the distribution chain of a medical sensor node may not be fully trusted by the end user: the devices could come out of the hands of different manufacturers and users. This rules out the first two types of predistribution methods in traditional WSNs, that is, there will not exist shared keys or common security context within the IMDs before they arrive at end users. Second, a BAN is often formed in an ad hoc way with unpredictable topology, while "plug-n-play" is the ideal usability goal. It is hard for the users to distribute keys manually since they usually are not experts. Most existing works on user-aided key predistribution in WSNs involve cumbersome human efforts [Kuo et al. 2007; Law et al. 2010] and are not very user-friendly. Third, a central root of trust or a PKI would

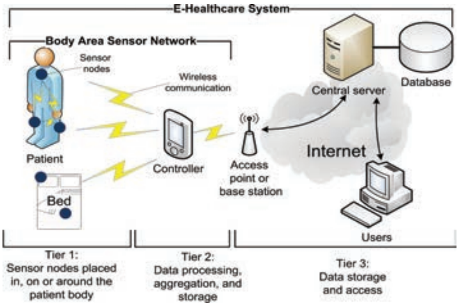

E-Healthcare System
Body Area Sensor Network
Sensor nodes
Wireless communication
Patient
Bed
Controller
Access point or base station
Central server
Database
Internet
Users
Tier 1: Sensor nodes placed in, on or around the patient body
Tier 2: Data processing, aggregation, and storage
Tier 3: Data storage and access

Fig. 1. A typical body area network and its relationship with the e-healthcare system.

be impractical for BANs, not only because they require costly infrastructure but also due to the high complexity involved in the revocation of nodes.

This gives rise to the problem of secure ad hoc initial trust establishment for a BAN, which happens before the BAN is actually deployed. Here we highlight several key differences between this and traditional key predistribution in WSNs. (1) Since secret keys are not assumed to be predistributed, trust must be established despite the lack of a common security context, and no central trusted parties can be the root of trust except that the user trusts herself. In particular, in practice, a group of BAN devices must be correctly associated with the intended patient, lest the wrong medical data be collected. This requires the IMDs to be authenticated to each other and to the BAN controller, which forms the group securely. Secret keys which can belong only to the intended group should be generated. (2) The traditional authentication goal [Bellare and Rogaway 1994] only stipulates that each participant is assured that each message appears to come from the true identity that generated it. However, in a BAN, since the wireless communication cannot be perceived by a human, in addition to traditional authentication, it is desirable to let a human user physically make sure that the devices ultimately authenticated to each other include and only include the intended devices that s/he wants to participate, which is often referred to as *demonstrative identification* [Chen et al. 2008; Lin et al. 2009] in usable security. To achieve this, the mechanism should be user-friendly, that is, involving as few human interactions as possible. (3) BAN applications are usually time critical, which mandates the trust bootstrap process to be fast and scalable. For instance, in EMS, an additional five minutes delay may result in a difference between life and death. Of course, overhead is an important concern since the medical sensor nodes are extremely resource constrained.

A unique challenge is that a secure communication channel shall be established out of an insecure channel for all the BAN devices upon their first meet, since IMDs communicate through wireless. This can be achieved by the so called *secure device pairing* concept that pairs up two devices [Li et al. 2010b]. A straightforward solution is to apply device pairing between the controller and each of the $N - 1$ IMDs to establish individual keys, based on which the pairwise keys and group key can be derived. However, this requires about $N - 1$ human interactions, with each one needing tens of seconds. Many current device pairing techniques are designed for pairing only two devices, which will require many runs for a BAN. Many others are unsuited for IMDs with limited resources and little human interface. GAnGS [Chen et al. 2008] is an exception, but it still requires N interactions.

In this article, we propose the *group device pairing* (GDP) protocol that establishes shared secret keys within a BAN out of nothing, that is, it relies on neither prior shared secrets nor common measurements nor a PKI. GDP sets up an authenticated BAN group (including a shared group key and individual secret keys among devices) with much fewer human interaction (constant) than establishing authenticated individual shared keys between the nodes one at a time using traditional device pairing techniques. In GDP, each device authenticates itself to every other device in the group as a legitimate member, which can be verified visually by a human. With the initial shared secret keys, standard cryptographic methods can be applied to generate other secret keys on demand after BAN deployment.

1.1. Our Contributions

We propose a suite of novel schemes for secure ad hoc initial trust establishment and key management in BAN.

(1) We put forward GDP as the primary scheme for initial trust establishment that relies on zero prior security context. GDP is essentially a user-aided multi-party authenticated key agreement protocol which combines the concept of device pairing and group key agreement in an unique way. We propose to use simultaneous comparison of synchronous LED blinking sequences on multiple resource-constrained devices by human users as an auxiliary out-of-band (OOB) channel to authenticate the key exchange in the group. An authenticated group key and individual shared secret keys among IMDs can be set up for a batch of BAN devices only in one shot. As a secondary scheme, we also propose a pairwise device pairing (PDP) protocol which establishes a shared symmetric secret key between a controller and an IMD without relying on key predistribution. The GDP is particularly suitable for BAN, because it typically contains less than 100 IMDs and the devices are within one-hop range.

(2) GDP enables efficient key management after network deployment. Multiple types of keys can be derived on-demand based on the initial keys obtained during trust establishment before deployment. Also, dynamic operations, such as regular key updates, batch node addition, and revocation are supported naturally by GDP. Our scheme is mostly based on symmetric key cryptography (SKC), thus having low communication and computation overhead.

(3) We formally prove the security of both schemes (GDP and PDP) based on the Bellare-Rogaway model [Bellare and Rogaway 1994] and give the security guarantees under the existence of a computational bounded adversary. The distinct features of our protocols and security proofs compared with other existing ones are the following. (1) Many previous protocols either require the use of non-malleable commitment schemes that involve heavy public key cryptography (PKC), or their security has not been formally proven. In contrast, our GDP and PDP both adopt commitment schemes that can be efficiently constructed from hash functions, while we prove their security without depending on the non-malleability of the commitments. (2) Our GDP protocol is also secure against compromised insider nodes with the fewest communication rounds, while the only assumption underlying that is minimal, that is, having a non-compromised controller.

(4) We carry out a thorough efficiency analysis for GDP and implement it on a ten node sensor network testbed to evaluate its performance. Experimental results show that initial trust establishment can be done within 30 seconds with low overhead in terms of time and energy consumption. GDP is secure yet practical. To the best of our knowledge, we are the first to propose, implement, and test the feasibility of the visual OOB channel based on human comparison of simultaneous LED blinking patterns.

1.2. Related Works

The problem of secure initial trust establishment in BANs has received little attention so far. Most previous works focus on security issues such as key management [Lorincz et al. 2004; Morchon et al. 2006; Malasri and Wang 2007], encryption [Lorincz et al. 2004; Malasri and Wang 2007; Tan et al. 2008], and access control [Tan et al. 2008]. However, it is a non-trivial issue to securely establish a secure communication channel among a BAN and associate it to the correct patient before any data communication happens.

1.2.1. Biometrical Methods. Biometrical values [Poon et al. 2006; Venkatasubramanian and Gupta 2010; Venkatasubramanian et al. 2010; Singh and Muthukkumarasamy 2007] have been used to establish a secure channel from which nodes can derive a common secret that associates the BAN to a specific patient's body. For example, electrocardiogram (EEG) and photoplethysmogram (PPG) has been exploited [Poon et al. 2006; Venkatasubramanian and Gupta 2010; Venkatasubramanian et al. 2010]. This realizes initial trust establishment in a plug-and-play manner. However, it requires specific hardware for all the nodes to be equipped with the same sensing capability. Moreover, this biometrical channel is not always available since it does not apply to sensor devices that are not placed on the human body, for example, those that monitor the surrounding environment.

1.2.2. Key Generation Based on Channel Characteristics. Mathur et al. [2008] proposed to extract a secret key between two wireless devices out of an unauthenticated wireless channel using a received signal strength indicator (RSSI). Jana et al. [2009] evaluated the effectiveness of key extraction methods using RSSI in real environments. These methods do not rely on key predistribution, but the key generation rate is limited by the wireless channel and currently group key generation is not enabled.

1.2.3. Key Predistribution in BAN. Recently, the trust establishment in BAN was studied by Keoh et al. [2009] under the context of secure sensor association. Each sensor node is associated with the controller one by one, using public-key based authentication, where a user compares LED blinking patterns to verify each association. However, their scheme assumes the existence of a trusted authority (TA) and still relies on the predistribution of public keys onto the sensor nodes. Also, it does not support batch deployment. In "message-in-a-bottle" [Kuo et al. 2007] and KALwEN [Law et al. 2010], a closed faraday-cage is employed as a secure channel in which keying materials are predistributed to all the intended sensor nodes before deployment. Secure sensor association is achieved in the sense that the user is assured no attackers out of the cage can associate with the same patient. However, costly additional hardware is required and it is cumbersome to add new nodes.

1.2.4. Secure Device Pairing. Device pairing is a promising technique for generating a common secret between two devices that shared no prior secrets with minimum or without additional hardware. It employs some low-bandwith out-of-band (OOB) channel to aid the authentication of information exchanged in the insecure wireless channel. Most proposed OOB channels rely on some form of human user participation. Well-known examples include the "resurrecting duckling" [Stajano and Anderson 2000], "talking-to-strangers" [Balfanz et al. 2002], "seeing-is-believing" [McCune et al. 2005], Loud-and-clear [Goodrich et al. 2006], and short string comparison based key agreement schemes [Cagalj et al. 2006; Pasini and Vaudenay 2006]. The usability of device pairing protocols based on various OOB channels is also evaluated [Nithyanand et al. 2010; Kumar et al. 2009]. For a comprehensive survey, please refer to [Nguyen and Roscoe 2011].

1.2.5. Group Message Authentication Protocols. The idea of user-aided authentication has also been adopted in group message authentication protocols, where each group member wants to transfer an authenticated data copy from her device to the other's.

For example, GAnGS [Chen et al. 2008] requires $O(N)$ human interactions and also uses digital signatures, which increase computational complexity. In SPATE [Lin et al. 2009], this is done through comparing T-flags. Each group member carries out N comparisons in parallel to authenticate other members' data. However, SPATE is specifically designed for message exchange and is not for group key agreement, and it lacks a formal security proof. Laur and Pasini [2008] proposed a group message authentication and key agreement protocol (SAS-GAKA) based on comparison of short authentication strings (SAS). However, it does not achieve group demonstrative identification. Moreover, SAS and T-flags are not applicable for sensor nodes because they require richer device interfaces. Therefore, none of SPATE and SAS-GAKA is suitable for secure, fast, efficient, and user-friendly initial trust establishment in BANs. In GDP, the whole group is authenticated and the group key is generated in one shot (i.e., requires one-time visual comparison of synchronized LED blinking patterns).

The most recent work that is close to ours is GAP [Perković et al. 2011]. GAP is a user-aided group message authentication protocol that can be applied to wireless sensor networks. It also exploits the idea of synchronous LED blinking pattern as the OOB channel. The authors also discussed how to deal with semi-authenticated visual light channels, which is orthogonal to our contribution. However, the security of GAP requires the use of non-malleable commitment schemes, where known constructions are much more inefficient than the hash commitments used in this article.

2. PROBLEM DEFINITION

2.1. Network Model

A BAN consists of a controller (gateway node) and a group of IMDs (medical sensor nodes). The size of the network varies, which may range from a few to the order of hundreds. Although the IMDs could be heterogenous in functionalities, we assume they are equipped with low-end, form-factor sensor nodes (e.g., comparable with Tmote). To meet the interoperability requirement, all of them are equipped with the same wireless communication interface, say ZigBee, and so is the controller. The sensors are limited in energy, communication, processing, and storage capabilities, while the energy and computation resources of the controller are more ample.

The sensors may be placed in, on, or around the patient's body. Although there is no consensus on the communication technologies in a BAN, the communication ranges in most current proposals are larger than 3 m (e.g., ZigBee). This is enough to assure that all nodes can be reached in one hop after deployment. Hence, we will assume a star topology. Each BAN has a patient who may be regarded as its owner, as well as a user who sets up the network. The latter is often a nurse but may also be the patient.

2.2. Design Requirements

2.2.1. Security Goals. The initial trust establishment during predeployment should establish a group key and/or individual keys shared between each sensor and the controller, which can be used for the controller to securely broadcast messages to the BAN later, such as queries. For the design of the PDP and GDP (user-aided authenticated key agreement protocols), we have the following security goals.

(1) *Key secrecy and key confirmation* [Ateniese et al. 2000]. For key secrecy, each group member should be assured that no non-member can obtain the group key. Key confirmation means that each member is assured that the peers actually possess the same key.

(2) *Group demonstrative identification*. Suppose that a set $\mathcal{G}$ of devices is intended by the user to be the group associated with a specific patient. If a group formation

process causes the set $\mathcal{G}'$ of devices to derive the same group key, then the user should be able to physically verify that $\mathcal{G}$ and $\mathcal{G}'$ are the same set.

Actually, this includes two properties: (1) key authenticity or consistency: each legitimate group member derives the same group key. If it also obtains individual shared keys, it must be assured that those keys come from the claimed true identities; (2) exclusiveness: the group includes only legitimate members and no attackers. This extends the "demonstrative identification" [Balfanz et al. 2002; McCune et al. 2005] but is different from PAALP in GAnGS [Chen et al. 2008].

In addition, for the key management after deployment, it should have *backward secrecy*, that is, a new group member should not learn about group keys in the past, and *forward secrecy*, that is, a former group member should not discover subsequent group keys for existing members. The session keys may include pairwise keys shared between pairs of sensor nodes so that they can securely distribute their data to other sensors. Sometimes, cluster keys are also needed in BANs.

2.2.2. Usability Goals

(1) *Efficiency*. A BAN often consists of low-end devices, relies on battery energy, and is intended to last at least for several days [Hanson et al. 2009; Lorincz et al. 2004; Jovanov et al. 2005]. To match the low capabilities of the sensors in BAN and to minimize energy consumption, it is important to minimize computation, communication, and storage overhead. Therefore, expensive cryptographic functions such as public-key operations should be avoided whenever possible.
(2) *Fast operation and user friendliness*. The initial trust establishment in a BAN should be fast while involving as few and intuitive human interactions as possible. Especially, batch deployment of devices should be supported.
(3) *Error proof*. Since humans make mistakes, the procedure must be easy to follow. Also, the system should be able to detect errors or attackers and alert the user.
(4) *Requires no additional hardware*. In order to reduce the cost of the system, it is essential to use commercial-off-the-shelf (COTS) products and to use fewer hardware components. For example, there should be no auxiliary devices. Also, the sensors usually do not have physical interfaces such as USB, because they may constrain form factors.

In addition, because the devices may be manufactured by different vendors which are hard to interoperate, we assume there are no preloaded public keys, certificates, or pre-shared secrets among the devices in a BAN. The sensors are used in a plug-and-play manner.

2.3. Attack Model

The attacker can either be an outsider or insider. An outsider does not compromise any devices in the intended BAN group, while an insider can compromise any of the sensor device. The attacker is able to eavesdrop, intercept, modify, replay, or inject the wireless communication between any devices in range. The attacker can also compromise a certain number of sensor nodes after deployment.

The main goals of an attacker are to obtain the secret keys by eavesdropping, impersonate as a legitimate group member to join the group, prevent one or more legitimate group members to join the group, act as the man-in-the-middle and try to split the intended group into two or more subgroups, maliciously modify the information contributed by legitimate group members so as to violate key authentication, and disrupt the group. The attacker can also pose as multiple identities to join the group, which is a Sybil attack. We do not consider denial of service (DoS) attacks in this article.

Table I. Frequently Used Notations

$H()$	A cryptographic hash function
$\mathcal{H}(m, r)$	Digest function with input m and key r
$x \leftarrow RS, x \in_R S$	Choose x uniformly from set S
$E_K\{\cdot\}$	Symmetric encryption with key K
$\hat{x}$	The unauthenticated version of x
$a\|b$	Concatenation of a and b
M_i	The ith group member
$\mathcal{G}$	The group of devices intended to associate to a patient
K_G, K_{ij}	The group key, the pairwise key between nodes i and j
$\mathcal{S}_k$	A subgroup of index k
N	Total number of devices in the group
$\mathbb{Z}_q^*$	Multiplicative group of prime order q
$\mathbb{F}_p$	Finite field of size p
n	The length of nonces
ℓ	Length of the short authentication string

We assume only that the controller is not compromised during the initial trust establishment process (i.e., is trusted by the user)[1] This is because the user can recognize his/her controller by password, and the controller is usually better kept and protected. Note that devices do not trust each other before the initial trust establishment.

3. BACKGROUND, NOTATIONS AND DEFINITIONS

3.1. Communication Channels in Device Pairing

In this article, we consider secure device pairing protocols (or user-aided authentication protocols) with multiple communication channels. Usually there are two kinds of channels: one is the normal *Dolev-Yao* channel, the other is an auxiliary *out-of-band* (OOB) channel. In a Dolev-Yao channel, all the messages transmitted between two devices can be overheard, deleted, or modified by the adversary. Examples may include the wireless channel. In an OOB channel considered in this article, messages cannot be modified or delayed from one session to another. The definition of the OOB channel corresponds to the empirical channel defined in Nguyen and Roscoe [2011], and can be regarded as authentic. The OOB channel is usually bandwidth-limited, as compared with a Dolev-Yao channel. The former is represented as "$\leftrightarrow$" in this article, while the latter is denoted as "$\Longleftrightarrow$".

Practical factors need to be considered when choosing the type of OOB channel in a device pairing protocol. In a BAN, sensor nodes may only have LED lights, beepers, and buttons, but no interfaces, such as camera, displays, or keyboards; yet the controller may have all of them. Under this asymmetric setting, the methods in McCune et al. [2005] and Balfanz et al. [2002] are unable to achieve mutual authentication. Fortunately, the Blink-Blink (BB) pairing method proposed in Prasad and Saxena [2008] was shown to be a practical approach. Briefly, both devices encode a short authentication string (SAS) obtained from a protocol run to a synchronized LED blinking pattern, where a '1' bit encodes to a "blink" (on) period and a '0' bit encodes to an "off". Then the user compares the patterns and accepts the results if they are the same. This is essentially a visual OOB channel between two devices, and we extend it to multiple devices in this article.

[1] In the preliminary version of this article [Li et al. 2010], we assumed all the devices to be benign during the predeployment phase, so the current assumption is much weaker.

3.2. Commitment Schemes

Commitment schemes are important cryptographic primitives that have been widely used in message authentication [Laur et al. 2005] and authenticated key agreement protocols [Cagalj et al. 2006; Pasini and Vaudenay 2006; Laur and Pasini 2008]. Typically, a commitment scheme consists of two algorithms.[2]

—Commit($INFO, x$) → (c, d), where $INFO$ is public data, x is n-bit private data, c is the commitment value, and d is an opening value. The algorithm is probabilistic.
—Open($INFO, c, d$) → $x \in \{0, 1\}^n \cup \{\perp\}$, which outputs the committed value x. If c is not a valid commitment, then it returns $\perp$. This algorithm is deterministic, and correctness implies that for any $x \in \{0, 1\}^n$, Open($INFO$, Commit($INFO, x$)) $= x$.

A commitment scheme should have two basic properties: *hiding* and *binding*. Their definitions are as follows.

Definition 1 (*(ϵ_h, T_h)-Hiding*). Given $(c, INFO)$, the probability that an adversary can correctly guess the value of x before the opening value d is revealed is upper bounded by ϵ_h in time T_h.

Definition 2 (*(ϵ_b, T_b)-Binding*). The probability that an adversary can open a commitment value c to a different x' afterward the one committed by c is upper bounded by ϵ_b in a time T_b.

In many existing user-aided authentication protocols [Perković et al. 2011; Laur and Pasini 2009; Laur and Nyberg 2006; Vaudenay 2005; Laur and Pasini 2008], the commitment schemes used are required to have a third property, *non-malleability*, which is stronger than the preceding basic ones. However, non-malleable commitment schemes are usually very inefficient in practice [Laur et al. 2005; Laur and Nyberg 2006], which will be unsuitable for low-end sensor nodes like Tmote. Fortunately, as we will show later, this property is not necessary for provable security of our proposed protocols. We instantiate the commitments using the following efficient construction from Pass [2003] based on a cryptographic hash function.[3]

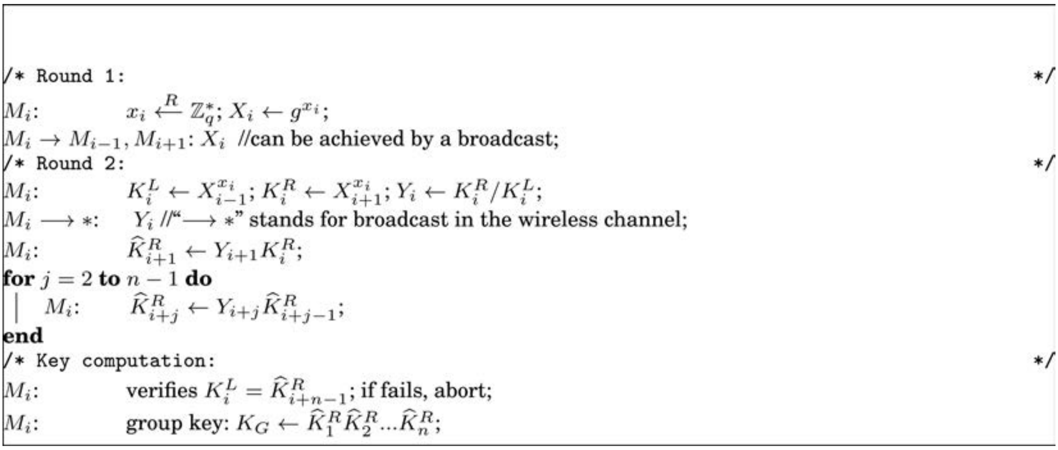
Definition 3 (*Hash Based Commitment Scheme*). Assume we have a cryptographic hash function H that can be modeled as a random oracle: $\{0, 1\}^{2n} \to \{0, 1\}^{l(n)}$, where $l(n) \leq \text{poly}(n)$. Then we have the following scheme.

—Commit. Given x, randomly pick $r \leftarrow \{0, 1\}^n$ and compute $c = H(x, r)$.
—Open. Let $d = (x, r)$. Output x if $c = H(x, r)$.

This scheme achieves hiding and binding [Pass 2003]. To commit to a longer message x, we can first hash it to n bits using a collision-resistant hash function and then commit, which is a general method [Halevi and Micali 1996]. Therefore, with public data $INFO$ and a message to be committed (m), we can set $x = INFO|m$, while the hiding and binding properties defined in Definitions 1 and 2 still hold. We will denote the hash commitment using HCommit and HOpen.

[2] In this article we adopt the definition from Nguyen and Roscoe [2011].

[3] In a few previous user-aided message authentication protocols, one-way hash functions (OHF) have been adopted as a practical alternative for commitment schemes [Zimmermann et al. 2006; Alliance 2006; Lin et al. 2009]. But to the best of the authors' knowledge, there have been no formal security proofs for such protocols up to date. In Laur and Pasini [2009], a security proof was posed as an open problem. We here provide security proofs for our protocols.



```
/* Round 1:                                                              */
$M_i$:        $x_i \xleftarrow{R} \mathbb{Z}_q^*$; $X_i \leftarrow g^{x_i}$;
$M_i \rightarrow M_{i-1}, M_{i+1}$: $X_i$ //can be achieved by a broadcast;
/* Round 2:                                                              */
$M_i$:        $K_i^L \leftarrow X_{i-1}^{x_i}$; $K_i^R \leftarrow X_{i+1}^{x_i}$; $Y_i \leftarrow K_i^R / K_i^L$;
$M_i \longrightarrow *$:   $Y_i$ //"$\longrightarrow *$" stands for broadcast in the wireless channel;
$M_i$:        $\widehat{K}_{i+1}^R \leftarrow Y_{i+1} K_i^R$;
for $j = 2$ to $n - 1$ do
    $M_i$:    $\widehat{K}_{i+j}^R \leftarrow Y_{i+j} \widehat{K}_{i+j-1}^R$;
end
/* Key computation:                                                      */
$M_i$:        verifies $K_i^L = \widehat{K}_{i+n-1}^R$; if fails, abort;
$M_i$:        group key: $K_G \leftarrow \widehat{K}_1^R \widehat{K}_2^R ... \widehat{K}_n^R$;
```

Fig. 2. Unauthenticated DB key agreement protocol ($1 \leq i \leq N$).

3.3. Digest Functions

In this article, we will make use of a digest function proposed by Nguyen and Roscoe [2008, 2011]. The digest function is defined as a mapping.

Definition 4 (*Digest Function*). $\mathcal{H}(m, k)$: $\{0, 1\}^L \times \{0, 1\}^n \rightarrow \{0, 1\}^\ell$ is a mapping where m is the message to be digested and k is the key. It shall have two properties.

(1) (ϵ_u-key-based uniformity) for any fixed m and y, $\Pr_{k \in_R \{0,1\}^n}[\mathcal{H}(m, k) = y] = \epsilon_u$.
(2) (ϵ_r-no uniform compensation) for any fixed θ and $m \neq m'$, $\Pr_{k \in_R \{0,1\}^n}[\mathcal{H}(m, k) = \mathcal{H}(m', k \oplus \theta)] = \epsilon_r$.

The key-based uniformity says that upon varying the key k, the output of the digest function should be uniformly distributed. And no uniform compensation means there should not exist θ such that it can always compensate the change in the digest output incurred by a different m' than m, for any varying key k.

A concrete construction is given in Nguyen and Roscoe [2008] based on matrix product, where the ideal properties are achieved: $\epsilon_u = \epsilon_r = \frac{1}{2^\ell}$. Usually the output of a digest function is a short string, for example $\ell = 16$ bits. Note that it is similar to a universal hash function, but a universal hash usually concerns collision resistance with respect to the same key.

3.4. Group Key Agreement Scheme

A contributory group key agreement establishes a group key based on no pre-shared secret, where every member equally contributes one share of the group key. In this article, we choose the unauthenticated group key agreement protocol (UDB) proposed by Dutta and Barua [2008] as a primitive. It is based on the Diffie-Hellman (DH) key agreement and is provably secure and only requires two rounds of communication. However, its authenticated version uses digital signatures, which requires PKI and is unsuitable for BANs. We describe the UDB protocol for completeness in Figure 2. $\mathbb{Z}_q^*$ is a multiplicative group of prime order q, where g is a generator. Note that $K_G = g^{x_1x_2+x_2x_3+\cdots+x_nx_1}$. Each node broadcasts two messages and performs three modular exponentiations: two $N-2$ modular multiplications and one modular division.

4. SECURE AD HOC TRUST INITIALIZATION AND KEY MANAGEMENT FOR BAN

4.1. Overview

Conceptually, the working cycle of a BAN mainly consists of three phases: predeployment, deployment, and working phases. In the predeployment phase, the sensor nodes are bootstrapped for the first time after being purchased; thus, initial trust among sensors should be established in this phase. For this phase, we propose two schemes for securely establishing the initial shared secrets among a group of ad hoc BAN devices (including a controller and multiple sensors), without relying on any prior security context (or pre-shared secrets) among the devices. The core of the first scheme (Scheme I) is a pairwise device pairing protocol (PDP), also known as a user-aided two-party authenticated key agreement, where a human user aids the authentication process by verifying simultaneous LED blinking patterns on both devices. By running the PDP protocol between the controller and each sensor one by one, each sensor derives an individual symmetric secret key with the controller. After that, the group key and pairwise keys can be established. Scheme I's complexity is $O(N)$ in terms of human effort. To improve upon it, we propose the group device pairing (GDP) protocol, also known as user-aided multi-party authenticated key agreement. The GDP establishes authenticated group key and individual symmetric keys in a group of devices in one shot with $O(1)$ human effort. Pairwise keys can also be subsequently obtained based on those keys. Both schemes are security enhanced versions of the corresponding ones in the preliminary version of this article [Li et al. 2010]. In the GDP, the only additional assumption is that the controller is not compromised, which is reasonable since it is usually better protected by the human user. In the next section, we also prove the security of both PDP and GDP formally, while the GDP protocol is also secure against compromised sensor nodes inside the group.

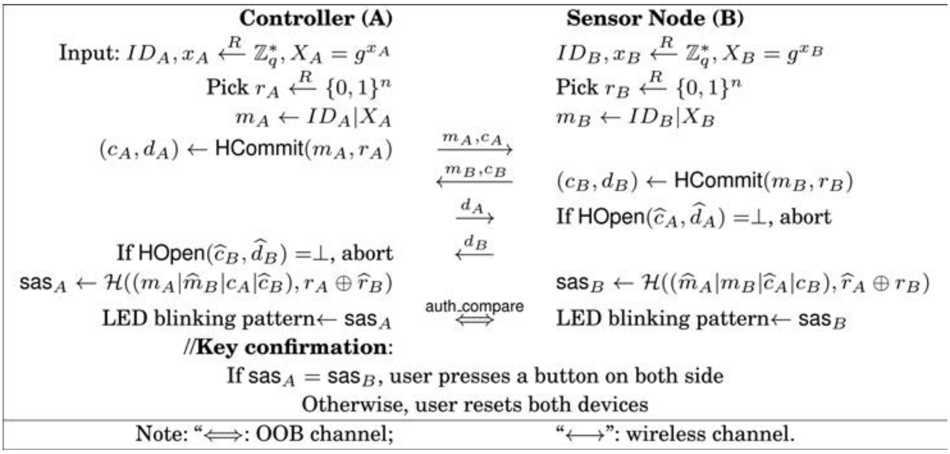
In the deployment phase, nodes are actually deployed to designated places on/in/around the human body. Neighbor discovery is performed to form a BAN topology, pairwise keys are actually computed, and a logical key hierarchy is established. For the working phase, the regular functions (e.g., collecting and reporting medical data) are executed. We then discuss periodical key updates and how to handle node join/leave/revocation operations efficiently.

4.2. Initial Trust Establishment via User-Aided Two-Party Authenticated Key Agreement

In the predeployment phase, a group of sensor nodes and a controller picked by the user must be uniquely and securely associated to the patient they will serve for. This is done through establishing initial secret keys, including individual keys and a group key. Rather than predistributing key materials onto each device beforehand (where the whole process may not be fully trusted), our approach is based on the concept of device pairing, which does not rely on any prior security context among nodes. In this section, we first present a straightforward scheme (Scheme I) where the controller establishes an individual secret key with each sensor one by one via our PDP protocol.

4.2.1. The Pairwise Device Pairing Protocol. The PDP is depicted in Figure 3. It is based on the DH key agreement and takes the DH public keys as part of the messages to be authenticated. The protocol essentially has three rounds, and the high-level idea can be described as "joint commitment before knowledge" [Nguyen and Roscoe 2011]: it means there is a point in every partial execution of the protocol such that both parties are committed to a value D (in our case, it is the SAS digest), but they do not yet know D, and in every successful completion of this partial execution, the parties are committed to the same value for D.

At first, A and B both generate a DH public value (X_A and X_B), and a random nonce (r_A, r_B), respectively. In the first round, they compute hash commitments (c_A, c_B) to



Controller (A)		Sensor Node (B)
Input: $ID_A, x_A \xleftarrow{R} \mathbb{Z}_q^*, X_A = g^{x_A}$		$ID_B, x_B \xleftarrow{R} \mathbb{Z}_q^*, X_B = g^{x_B}$
Pick $r_A \xleftarrow{R} \{0,1\}^n$		Pick $r_B \xleftarrow{R} \{0,1\}^n$
$m_A \leftarrow ID_A \vert X_A$		$m_B \leftarrow ID_B \vert X_B$
$(c_A, d_A) \leftarrow \mathsf{HCommit}(m_A, r_A)$	$\xrightarrow{m_A, c_A}$	
	$\xleftarrow{m_B, c_B}$	$(c_B, d_B) \leftarrow \mathsf{HCommit}(m_B, r_B)$
	$\xrightarrow{d_A}$	If $\mathsf{HOpen}(\widehat{c}_A, \widehat{d}_A) = \perp$, abort
If $\mathsf{HOpen}(\widehat{c}_B, \widehat{d}_B) = \perp$, abort	$\xleftarrow{d_B}$	
$\mathsf{sas}_A \leftarrow \mathcal{H}((m_A \vert \widehat{m}_B \vert c_A \vert \widehat{c}_B), r_A \oplus \widehat{r}_B)$		$\mathsf{sas}_B \leftarrow \mathcal{H}((\widehat{m}_A \vert m_B \vert \widehat{c}_A \vert c_B), \widehat{r}_A \oplus r_B)$
LED blinking pattern$\leftarrow \mathsf{sas}_A$	$\overset{\text{auth\_compare}}{\Longleftrightarrow}$	LED blinking pattern$\leftarrow \mathsf{sas}_B$
//**Key confirmation**:		
If $\mathsf{sas}_A = \mathsf{sas}_B$, user presses a button on both side		
Otherwise, user resets both devices		
Note: “$\Longleftrightarrow$: OOB channel;		“$\longleftrightarrow$”: wireless channel.

Fig. 3. User-aided two-party authenticated key agreement protocol (PDP) between the controller and a sensor node in Scheme I.

their corresponding nonces and IDs and exchange the messages m_A and m_B along with the commitments. In the second round, the decommitment values are exchanged which reveal the nonces to each other. The preceding two rounds exchange messages using the wireless channel. In the third round, A and B both compute an SAS in order to authenticate m_B and m_A, which is a digest based on their own and received messages and keys. The SASes are encoded into LED blinking patterns which are displayed synchronously over a visual OOB channel. The user compares the patterns (in an authenticated way) and accepts the authentication if they are the same. If authenticated, $K_{AB} = \widehat{X}_B^{x_A} = \widehat{X}_A^{x_B} = g^{x_A x_B}$. After that, the user needs to let both the controller and the sensor know the acceptance of the authentication result (key confirmation) by simply pressing a button on both devices.

There are some subtle points to be noticed. First, we have included the ID and DH public value of each party in its hash commitment. The ID is used to prevent the replay attack, where the adversary can copy a commitment of A and later deliver it to A again. And the inclusion of DH public value binds it with the commitment value, whose function will be more clear in the security proof. Second, we need to ensure a strict order of message exchange between the parties in order to synchronize both devices about the ending of phases. This can be done by announcing the devices' IDs before round I, and a node only sends its own data after receiving from the one with smaller ID. In the PDP, there is no constraint to the controller's ID. In contrast, we will see later in the GDP protocol that the controller's ID is required to be maximum. Third, in the SAS, we have included both parties' IDs, DH public values, and commitments, that is, the protocol transcript. This also turns out to be an important factor for the security of both PDP and GDP protocols. Finally, the key confirmation can only be done manually, because otherwise there will be man-in-the-middle attacks at this stage. For example, in the preliminary version of this article [Li et al. 2010], if the adversary establishes a different key with each of A and B before key confirmation, she will be able to deceive both A and B again at this stage.

4.2.2. Establishment of Group Key and Pairwise Keys. After $N - 1$ individual shared keys are established, a group key K_G is generated by the controller. To distribute the group key, the controller simply encrypts it $N - 1$ times using the individual shared keys and unicasts to each sensor node. Now the user enters the ID of the patient into the controller and associates the individual keys and the group key with this ID, which is also the ID of the BAN.

Next, in order to prepare for secure communication in the deployment phase and working phase, we need to distribute key materials to sensors so that they can establish pairwise keys afterwards. Here we use the Blundo's polynomial-based key pre-distribution method [Blundo et al. 1993]. The controller first randomly generates a bivariate t-degree symmetric polynomial $f(x, y) = \sum_{i,j=0}^{t} a_{i,j} x^i y^j$ defined over a finite field $\mathbb{F}_p$ with p being a large prime number.[4] The controller C (the group member with the largest ID, sometimes denoted as M_N) computes a univariate *polynomial share* for each node M_i (with ID i): $f_i(y) = f(i, y)$. Then it encrypts and unicasts this to each sensor node.

$$(msg1)\ C \longrightarrow M_i :\ i, E_{K_{Ni}}\{f_i(y)|MAC_{K_{Ni}}(f_i(y))\}, \tag{1}$$

where the message authentication code (MAC) provides authentication and integrity check, and K_{Ci} stands for the key shared between C and M_i. Now the pairwise key between i and j is $K_{ij} = f_i(j) = f_j(i) = K_{ji}$.

In addition, in order for the controller to authenticate itself afterwards, the controller generates a *one-way hash chain* [Lamport 1981], $\bar{k}_n, \bar{k}_{n-1}, \ldots, \bar{k}_0$, where $\bar{k}_i = H(\bar{k}_{i+1})$, $0 \leq i \leq n-1$. The controller distributes the commitment of the chain ($\bar{k}_0$) to all sensor nodes.

$$(msg2)\ C \longrightarrow M_i :\ E_{K_G}\{\bar{k}_0|MAC_{K_G}(\bar{k}_0)\}. \tag{2}$$

4.3. Initial Trust Establishment via User-Aided Multi-Party Authenticated Key Agreement

In Scheme I, associating sensor nodes one by one is very time consuming, since each pair of LED blinking requires tens of seconds. Therefore, a more scalable and efficient method must be developed. The GDP directly establishes initial secret keys in one shot, including a group key and individual keys among a group of devices through a multi-party authenticated key agreement. The idea is to authenticate the messages exchanged in a group key agreement scheme with a human user's help, that is, simultaneously comparing LED blinking patterns for a group of devices in an OOB visual channel.

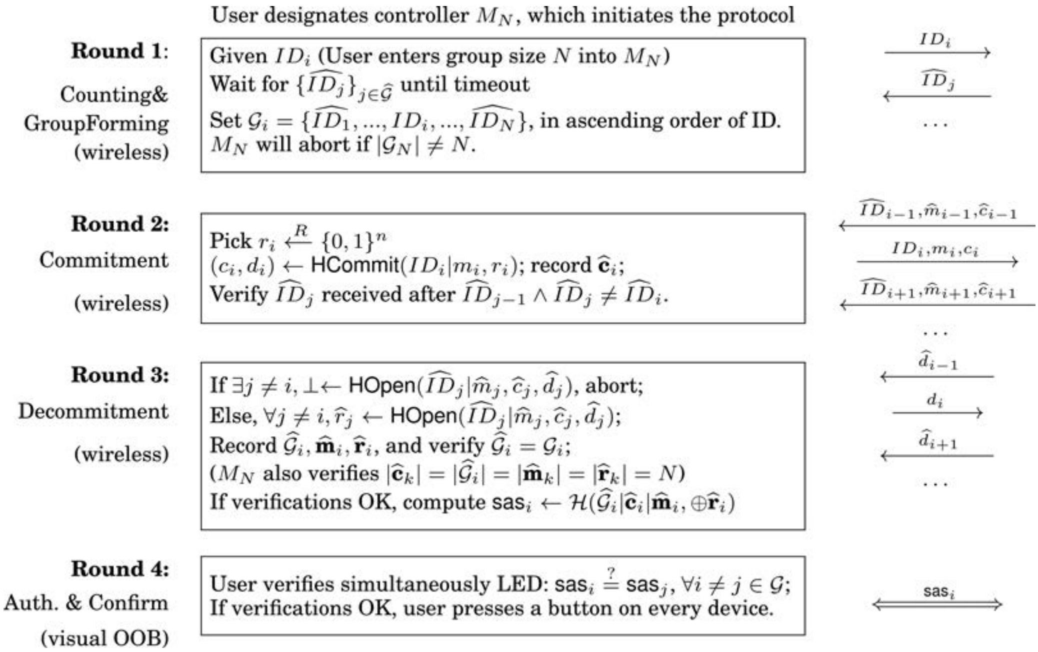
We first propose the core protocol, GDP. We present it in two steps: first we give a multi-party message authentication protocol (MP-MAP) and then build the GDP based on the MP-MAP. The MP-MAP adopts similar design principles with the underlying MAP protocol of PDP, and their protocol structures resembles each other.

4.3.1. The Proposed MP-MAP. The MP-MAP for a group $\mathcal{G}$ is outlined in Figure 4. It consists of four rounds. The first three rounds use wireless channel, while the fourth utilizes the visual channel.

Round 1 (wireless). In the counting and group forming phase, the user U would pick a group of N devices and place them in close proximity. She chooses the controller device M_N which has the largest ID among all devices (this can be ensured by assigning ID_N a very large number) enters the group member count (N) into M_N, and indicates to start the protocol. Each member device M_i broadcasts its own identity ID_i to the group and receives others' IDs. After a timeout, each M_i sorts the pool of IDs in ascending order and keeps its own view of the group $\mathcal{G}_i$. In addition, the controller checks if the group size equals to n; if not, it will abort. The true group is denoted as $\mathcal{G}$, which can be perceived by the user.

Round 2 (wireless). In the commitment round, each M_i generates a random nonce r_i as its own share of digest key to generate the SAS in the end. Then r_i is committed along

[4] For example, we can use $p \approx 2^{80}$ to provide an 80-bit symmetric key.



User designates controller M_N, which initiates the protocol

Round 1: Counting& GroupForming (wireless)

Given ID_i (User enters group size N into M_N)
Wait for $\{\widehat{ID_j}\}_{j\in\widehat{\mathcal{G}}}$ until timeout
Set $\mathcal{G}_i = \{\widehat{ID_1}, ..., ID_i, ..., \widehat{ID_N}\}$, in ascending order of ID.
M_N will abort if $|\mathcal{G}_N| \neq N$.

Round 2: Commitment (wireless)

Pick $r_i \overset{R}{\longleftarrow} \{0,1\}^n$
$(c_i, d_i) \leftarrow \mathsf{HCommit}(ID_i|m_i, r_i)$; record $\widehat{\mathbf{c}}_i$;
Verify $\widehat{ID}_j$ received after $\widehat{ID}_{j-1} \wedge \widehat{ID}_j \neq \widehat{ID}_i$.

Round 3: Decommitment (wireless)

If $\exists j \neq i, \perp \leftarrow \mathsf{HOpen}(\widehat{ID}_j|\widehat{m}_j, \widehat{c}_j, \widehat{d}_j)$, abort;
Else, $\forall j \neq i, \widehat{r}_j \leftarrow \mathsf{HOpen}(\widehat{ID}_j|\widehat{m}_j, \widehat{c}_j, \widehat{d}_j)$;
Record $\widehat{\mathcal{G}}_i, \widehat{\mathbf{m}}_i, \widehat{\mathbf{r}}_i$, and verify $\widehat{\mathcal{G}}_i = \mathcal{G}_i$;
(M_N also verifies $|\widehat{\mathbf{c}}_k| = |\widehat{\mathcal{G}}_i| = |\widehat{\mathbf{m}}_k| = |\widehat{\mathbf{r}}_k| = N$)
If verifications OK, compute $\mathsf{sas}_i \leftarrow \mathcal{H}(\widehat{\mathcal{G}}_i|\widehat{\mathbf{c}}_i|\widehat{\mathbf{m}}_i, \oplus\widehat{\mathbf{r}}_i)$

Round 4: Auth. & Confirm (visual OOB)

User verifies simultaneously LED: $\mathsf{sas}_i \overset{?}{=} \mathsf{sas}_j, \forall i \neq j \in \mathcal{G}$;
If verifications OK, user presses a button on every device.

$\xrightarrow{ID_i}$
$\xleftarrow{\widehat{ID}_j}$
...
$\xleftarrow{\widehat{ID}_{i-1}, \widehat{m}_{i-1}, \widehat{c}_{i-1}}$
$\xrightarrow{ID_i, m_i, c_i}$
$\xleftarrow{\widehat{ID}_{i+1}, \widehat{m}_{i+1}, \widehat{c}_{i+1}}$
...
$\xleftarrow{\widehat{d}_{i-1}}$
$\xrightarrow{d_i}$
$\xleftarrow{\widehat{d}_{i+1}}$
...
$\xLeftrightarrow{\mathsf{sas}_i}$

Fig. 4. Multi-party message authentication protocol (MP-MAP) at each device M_i. The message to be authenticated of each device is m_i.

with the message m_i and its ID, which are public data. Since the digest keys are hidden from the attacker in this round, all devices essentially have jointly committed to an SAS value that the attacker does not know. So the digest keys provide the randomness required for security. All devices send their commitments c_i in order, that is, ID_{i-1}'s transmission must precede that of ID_i's, and each device can verify this order. The purpose is to provide device synchronization, that is, they must agree on when one round ends. By using strict message ordering in rounds 2 and 3, the message sent by the device with the largest ID serves as the synchronization signal. It prevents possible attacks that exploit the desynchronization, for example the one discovered in Perković et al. [2011]. The controller will always be the last one to broadcast. Each device M_i also keeps record of the set of received $\widehat{c}_j$s—$\widehat{\mathbf{c}}_i = \{\widehat{c}_1, \ldots, c_i, \ldots, \widehat{c}_{N_i}\}$, where N_i should equal $|\mathcal{G}_i|$.

Round 3 (wireless). . In this round, each device M_i reveals its committed digest key by broadcasting the decommitment value so that others can verify the validity of the commitment and obtain $\widehat{r}_i$ (they will check if $\widehat{ID}_i, \widehat{m}_i, \widehat{r}_i$, and $\widehat{c}_i$ are a valid message-commitment pair). The controller, upon collecting all the other devices' commitments and digest keys, checks if the numbers of group members, commitments, messages, and digest keys all equal N (the controller is assumed to be not compromised). In addition, every other device should check the consistency of the group IDs with respect to $\mathcal{G}_i$ collected at the beginning. After that, the SAS is computed at each M_i as a digest of the protocol transcript, with the XOR of M_i's received set of $\widehat{\mathbf{r}}_i$ as digest key.

Round 4 (visual OOB). . This round is when most of the human efforts takes place. Next, the SASes are encoded into synchronized LED blinking patterns for user comparison. The duration of the LED blinking depends on the number of bits of the SAS. Usually 16–20 bits are enough for security. If all the patterns are the same, U confirms that authentication succeeded by pressing a button on every device.

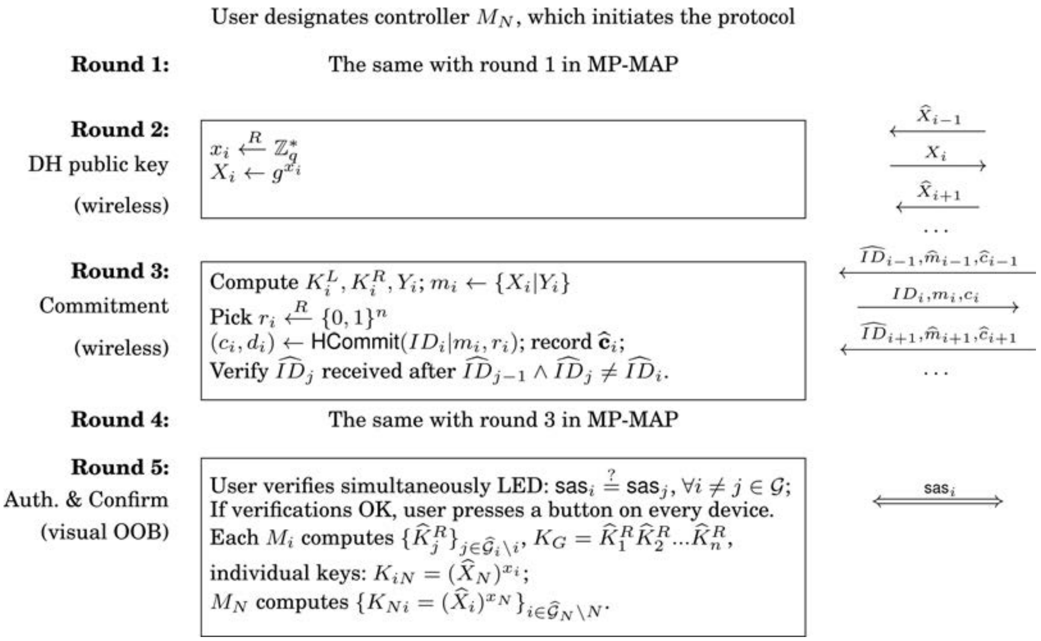

User designates controller M_N, which initiates the protocol
Round 1:
The same with round 1 in MP-MAP
Round 2:
DH public key
(wireless)
$x_i \xleftarrow{R} \mathbb{Z}_q^*$
$X_i \leftarrow g^{x_i}$
$\widehat{X}_{i-1}$
X_i
$\widehat{X}_{i+1}$
...
Round 3:
Commitment
(wireless)
Compute K_i^L, K_i^R, Y_i; $m_i \leftarrow \{X_i|Y_i\}$
Pick $r_i \xleftarrow{R} \{0,1\}^n$
$(c_i, d_i) \leftarrow$ HCommit$(ID_i|m_i, r_i)$; record $\widehat{\mathbf{c}}_i$;
Verify $\widehat{ID}_j$ received after $\widehat{ID}_{j-1} \wedge \widehat{ID}_j \neq \widehat{ID}_i$.
$\widehat{ID}_{i-1}, \widehat{m}_{i-1}, \widehat{c}_{i-1}$
ID_i, m_i, c_i
$\widehat{ID}_{i+1}, \widehat{m}_{i+1}, \widehat{c}_{i+1}$
...
Round 4:
The same with round 3 in MP-MAP
Round 5:
Auth. & Confirm
(visual OOB)
User verifies simultaneously LED: $\mathsf{sas}_i \overset{?}{=} \mathsf{sas}_j, \forall i \neq j \in \mathcal{G}$;
If verifications OK, user presses a button on every device.
Each M_i computes $\{\widehat{K}_j^R\}_{j \in \widehat{\mathcal{G}}_i \setminus i}$, $K_G = \widehat{K}_1^R \widehat{K}_2^R ... \widehat{K}_n^R$,
individual keys: $K_{iN} = (\widehat{X}_N)^{x_i}$;
M_N computes $\{K_{Ni} = (\widehat{X}_i)^{x_N}\}_{i \in \widehat{\mathcal{G}}_N \setminus N}$.
sas_i

Fig. 5. The multi-party key agreement protocol (GDP) at each device M_i. It establishes a group key and sensors' individual keys with the controller.

4.3.2. The Group Device Pairing Protocol. Next we describe the GDP protocol, outlined in which combines the MP-MAP and the UDB group key agreement protocol. Round 1 is the same as Figure 5, that in MP-MAP. In round 2, a Diffie-Hellman (DH) public key (X_i) is computed at each device and is exchanged among all the devices in the group. In round 3, each device first computes its Y_i value based on X_js received in round 2, and then takes $X_i|Y_i$ as the message m_i to be authenticated. Devices compute and exchange hash commitments in this round as in MP-MAP. Round 4 is the same as round 3 in the MP-MAP, which reveals the digest keys. Finally, in round 5, after confirming all the LED blinking patterns match, each device computes a group key based on all the previously received X_js and Y_js which should be already authenticated up to this point. In addition, as a byproduct, each sensor computes its individual key shared with the controller using the DH public key, and vice versa. As we will show in the next section, the GDP achieves almost the same level of security as the PDP, with the same SAS length. Therefore, using the same amount of human effort as in the PDP, an authenticated group key and individual keys are all established.

4.3.3. Initial Trust Establishment via GDP. Now we describe some practical issues, for example, how the GDP is applied to initial trust establishment in the BAN (also called secure sensor association). In reality, there is usually a limit to the number of LED blinking devices a human user can watch at the same time. We refer to this limit as N_{max}. If the number of the intended group of devices for a BAN $N = |\mathcal{G}| \leq N_{max}$, the user carries out one GDP for $\mathcal{G}$ to set up the group key K_G. If $N > N_{max}$, the user randomly picks nodes from $\mathcal{G}$ in a batch to form smaller subgroups whose sizes are equal to N_{max} whenever possible. The GDP protocol is then executed for each subgroup $\mathcal{G}(k)$. The controller must be in every subgroup so that it can establish a subgroup key $K_{G(k)}$ with each of them through GDP. When the last subgroup has only one sensor node left, Scheme I is automatically used to establish a pairwise key (however, it makes little

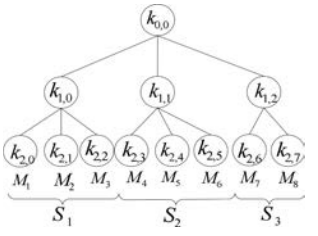


Fig. 6. A logical key tree for a BAN of nine nodes ($N_{max} = 3$). A key is indexed by its level λ and branch number μ. $\mathcal{G}(k)$ refers to a subgroup.

difference to the user). After that, the controller generates the final group key K_G and broadcasts it using encryption to each subgroup, $E_{K_{G(k)}}\{K_G|\mathcal{G}|MAC_{K_{G(k)}}(K_G|\mathcal{G})\}$, where $\mathcal{G} = \cup_k \mathcal{G}(k)$ and $|\mathcal{G}| = N$.

After the sensor association is successfully done, the group of devices need to set up the pairwise keys among them. There are two options. The simplest way is to reuse the DH public keys and let each $M_i, i \in \mathcal{G}$ compute $K_{ij} = (X_j)^{x_i}, \forall j \in \mathcal{G} \setminus i$. But this incurs additionally $N - 2$ exponentiation operations for each sensor device (except the individual key computation), which is not desirable for resource-constrained sensors. The other way is to use the method in Section 4.2.2, that is, let the controller broadcast material to each sensor which is encrypted under the sensor's individual key. And then each sensor computes the shared pairwise keys with others on its own. In this way, exponentiation operations are replaced with less costly field multiplication operations.

4.4. Deployment and Thereafter

The deployment phase establishes the pairwise and logical keys. Upon deployment, each node M_i first performs neighbor discovery. For each neighbor M_j, M_i computes the pairwise key K_{ij} as previously mentioned. In practice, in order to save storage space, a node can merely store the pairwise keys that it uses frequently, while computing the other pairwise keys on demand.

Then, the logical keys are derived naturally from the subgroup keys in GDP, which are used to form a logical key hierarchy (LKH). The LKH [Wong et al. 1998] has been proposed to achieve efficient key revocation. Since the LKH is a balanced binary tree, the message overhead for key revocation is $O(log_2(N))$. However, it is not very efficient for batch node addition or removal.

To avoid this drawback, we use a constant depth ($d = 3$), variable branch, and balanced key tree (Figure 6). Each internal node stands for a logical key, and each leaf node corresponds to the individual key of a sensor node. So we have $k_{0,0} = K_G$ and $k_{2,i} = K_{C,i+1}$. The keys $k_{1,i} = K_{G(k)}$ are the subgroup keys derived in the end of GDP. The branch of the root $\mu_{0,0}$ equals the number of subgroups, while the branch of a second-level node is $\mu_{1,i} = |\mathcal{G}(i)|$. The controller C has the information of the entire key tree. Note that, no messages are needed to transmit the logical keys for the tree in our scheme.

Note that our scheme can be easily extended to BANs with cluster topologies, since we can predict which nodes will form a cluster and thereby a subgroup by looking at their functionalities. For example, the use of several sensor nodes connected to 30 motion sensors is reported in Van Laerhoven et al. [2002] to detect a patient's acceleration and gait. A simple clustered BAN topology is shown in Figure 7. Some nodes form clusters (e.g., M_4, M_5 and M_6, M_7, M_8), while others are independent with each other (M_1, M_2, M_3). In order to save energy, the controller directly communicates with cluster heads and non-clustered nodes. In this case, the cluster keys will be the logical keys

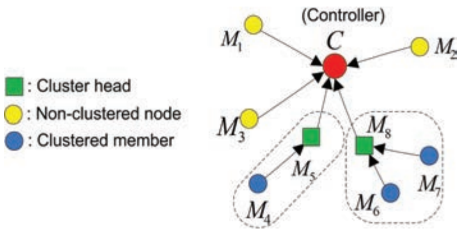


Fig. 7. A simple clustered BAN topology.

and the subgroup keys at the same time. We can use GDP to set up the cluster key for both clusters and use PDP to establish individual keys for each non-clustered node.

After that, the BAN is ready to function. In summary, now a sensor node M_i has the following key (material)s: K_G, $K_{i,N}$, $K_{G(k)}$, $f_i(y)$, $\bar{k}_0$. Since the keys may be compromised by cryptanalysis afterwards, we need to introduce sessions for the working phase that is, time periods across which keys are updated regularly. The preceding keys are all treated as keys in session 0. A key K in session i is denoted as $K(i)$.

4.4.1. Session Key Update. Periodically, the controller broadcasts an update message to the network. It is authenticated using the local broadcast authentication method [Zhu et al. 2003], since we assume the BAN is one hop. The controller first updates $f(x, y)$—$f_{i+1}(x, y) = f_i(x, y) + \Delta_{i+1}$, where $\Delta_{i+1} \overset{R}{\leftarrow} \mathbb{F}_p$. Then, it updates the logical keys as $k_{0,0}(i+1) = H(k_{0,0}(i))$, $k_{1,\mu}(i+1) = H(k_{1,\mu}(i))$, and broadcasts the following.

$$msg3 \leftarrow \text{“}\texttt{Update to session}\, i+1\text{”}|\Delta_{i+1},$$
$$C \longrightarrow * : E_{k_{0,0}(i)}\{msg3\}, \bar{k}_{i+1},\ MAC_{\bar{k}_{i+1}}(msg3).$$

Then, each sensor can authenticate C by verifying that $H(\bar{k}_{i+1}) = \bar{k}_i$.

Next, all sensor nodes update all the keys in its memory as the controller does. For the pairwise keys, node u computes $f_{u,i+1}(y) = f_{u,i}(y) + \Delta_{i+1}$. This achieves the update of all $\frac{N(N-1)}{2}$ pairwise keys through only one broadcast message.

4.5. Membership Management

4.5.1. Node Join. Adding one node is easy; we can just perform one device pairing using Scheme I. We will elaborate on how GDP supports efficient batch node addition.

Step 1. Before $l > 1$ new nodes join the BAN during session i, they are reset by the user (all dynamic memories are lost) and assumed to be benign.

Step 2. Before they are deployed, the same steps in GDP are performed by treating them as a new group, where the controller obtains the temporary group key K_G^T and all the logical keys.

Step 3. The controller advances the existing BAN to session $i+1$ without waiting until the end of session i. To this end, all nodes do the same thing as in a session key update.

Step 4. The controller predistributes new polynomial shares $f_{v,i+1}(y)$ for each new node v. Also, it encrypts $K_G(i+1)$ and $\bar{k}_{i+1}$ using K_G^T and broadcasts to the new nodes. A new key tree can then be derived that includes the new nodes. Then, the new nodes are deployed.

4.5.2. Node Leave/Revocation. Upon single-node leave or revocation during session i, the group key, logical keys, and pairwise keys are renewed to exclude the leaving node. The controller randomly generates a new group key $K_G(i+1)$. All the logical keys on the tree path of the leaving node are refreshed. For example, in Figure 6, say M_1 is revoked. Then, the controller sends the following messages.

$$\begin{aligned} C \rightarrow M_2 &: E_{k_{2,2}}\{k_{1,0}(i+1)\}; \\ C \rightarrow M_3 &: E_{k_{2,3}}\{k_{1,0}(i+1)\}; \\ C \rightarrow M_2, M_3 &: E_{k_{1,0}(i+1)}\{k_{0,0}(i+1)\}; \\ C \rightarrow M_4, M_5, M_6 &: E_{k_{1,1}(i+1)}\{k_{0,0}(i+1)\}; \\ C \rightarrow M_7, M_8 &: E_{k_{1,2}(i+1)}\{k_{0,0}(i+1)\}; \end{aligned}$$

where $k_{1,1}(i+1) = H(k_{1,1}(i))$, $k_{1,2}(i+1) = H(k_{1,2}(i))$. After that, the controller sends the updated polynomial share (Δ_{i+1}) to all nodes using authenticated broadcast. Thus, the revoked node cannot obtain the new group key and the updated polynomial share. It is straightforward to see how this is done when batch node leave event happens, for which we will analyze the efficiency in Section 6.

5. SECURITY ANALYSIS

For the authenticated key agreement (AKA) protocols in this article, there are essentially two security goals: key secrecy and key authenticity. A basic secrecy goal is defined with respect to a passive adversary, that is, an eavesdropper should have negligible advantage in deriving the shared key K_{AB}. In PDP, the only information sent over the wireless channel for the derivation of K_{AB} is the set of the X_is. Thus, key secrecy with a passive adversary amounts to that of a Diffie-Hellman key exchange, which follows from the assumption that the Decisional Diffie–Hellman (DDH) problem is intractable. In the GDP protocol, a similar passive secrecy guarantee follows from the secrecy of the UDB key agreement protocol [Dutta and Barua 2008].

Thus, key authenticity will be the AKA protocol security goal we study in the remainder of this section. The cores of our AKA protocols are their corresponding message authentication protocols (MAPs). In the following, we focus on defining and proving the security of MAPs. The security of an AKA protocol follows from the security of its underlying MAP and the security of the key agreement protocol against a passive adversary.[5]

Without loss of generality, we state the security definition of MAP using the multi-party scenario. Assume the group consists of N parties (devices), $\mathcal{G} = \{ID_1, ID_2, \ldots, ID_N\}$, for simplicity, we use i to represent ID_i. Each party $i \in \mathcal{G}$ has some message m_i to be authenticated to all the rest of the parties in $\mathcal{G}$, for example, in the PDP $m_i = \{ID_i, X_i\}$, while in GDP $m_i = \{ID_i, X_i, Y_i\}$.

Next, we define secure message authentication of an MAP based on the notion of "matching conversations" introduced by Bellare and Rogaway [1994] (details are provided in Appendix A). The following security definition captures the intuition that if a MAP is secure, then the only way that an adversary can make all parties accept at the end of a protocol run is to faithfully relay all the messages. We will use $\widehat{\mathbf{m}}_i$ to denote i's received vector (ordered set) of messages $\{\widehat{m}_{1i}, \ldots, \widehat{m}_{i-1i}, m_i, \widehat{m}_{i+1i}, \ldots, \widehat{m}_{Ni}\}$, and similarly $\widehat{\mathbf{c}}_i$ stands for the vector of received commitments by i, etc.

[5] To show this, the modular approach proposed by Bellare et al. [1998] can be applied. Specifically, It assumes two adversary models—the authenticated link model (AM) and the unauthenticated link model (UM). If a protocol is proven to be secure in the AM, then it can be shown to be secure in the UM provided that each message transferred between the parties is authenticated by a protocol called message transfer (MT) authenticator. In our setting, by saying "security of the key agreement protocol" we mean that its unauthenticated version (e.g., original Diffie-Hellman) should be secure in the AM, while the MAP can be regarded as an MT-authenticator.

Definition 5 (*Secure Message Authentication*). We say that Π is a (ϵ, T)-secure message authentication protocol with a group of participants $\mathcal{G}$ ($|\mathcal{G}| \geq 2$), if for any T-time adversary $\mathcal{A}$, the following hold.

(1) (Matching conversations $\Rightarrow$ acceptance). If all pairs of parties in $\mathcal{G}$ have jointly matching conversations, then all parties accept.
(2) (Acceptance $\Rightarrow$ matching conversations). Letting $\mathrm{Adv}_{\Pi}(\mathcal{A}) = \Pr[\textsf{All-accept} \wedge \textsf{No-Matching}^{\mathcal{A}}]$, where $\textsf{No-Matching}^{\mathcal{A}}$ refers to the event that the conversations are not jointly matching, we have $\mathrm{Adv}_{\Pi}(\mathcal{A}) \leq \epsilon$.

In condition (2), we may use the uncorrupted group $\mathcal{N} = \mathcal{G}$, in which case we speak of the adversary as an *outsider*. Alternatively, we may choose $\mathcal{N} \subsetneq \mathcal{G}$, and speak of an *insider* adversary. In a two-party MAP, one does not need to consider one of the parties being compromised, because then there is nothing to prove. Thus, we only discuss node compromise for the multi-party protocols.

5.1. Security of the PDP

We will refer to the message authentication protocol underlying the PDP as the two-party MAP (TP-MAP). We first state the following theorem.

THEOREM 1. *Assume that the digest function satisfies ϵ_u-key-based uniformity and ϵ_r-no uniform compensation. If the hash commitment scheme is (ϵ_h, T_h)-hiding and (ϵ_b, T_b)-binding, the TP-MAP is $(max\{\epsilon_u, \epsilon_r\} + \epsilon_h + 2\epsilon_b, 2T_b + T_h + O(1))$-secure.*

PROOF. Please refer to Appendix B. □

Security interpretation. The security levels achieved by the TP-MAP (and the MP-MAP, as we will see) depend mainly on the SAS's length ℓ. This is because the adversary's deception probability is dominated by either ϵ_u or ϵ_r, which should equal to $2^{-\ell}$ given an ideal digest function, while ϵ_h, ϵ_b reflect the security of hash commitment, which uses long nonces. Their values are approximately 2^{-n}, orders smaller than $2^{-\ell}$.

5.2. Security of the GDP

The MP-MAP can be proven as secure as the TP-MAP under the Bellare-Rogaway model, even when there exist compromised devices (insider attack). Our assumption is that the controller is not compromised, but any other sensor could be compromised by the adversary.[6]

THEOREM 2. *Assume that the digest function satisfies ϵ_u-key-based uniformity and ϵ_r-no uniform compensation. If the hash commitment scheme is (ϵ_h, T_h)-hiding and (ϵ_b, T_b)-binding, the MP-MAP is $(max\{\epsilon_u, \epsilon_r\} + \epsilon_h + 2\epsilon_b, 2T_b + T_h)$-secure.*

PROOF. Please refer to Appendix C. □

Remark. The MP-MAP and TP-MAP's security proofs are similar, and they both belong to the directly binding category [Nguyen and Roscoe 2011]. Interestingly, we can summarize several principles underlying both the multi-party and two-party version of the MAP protocol in this article. (1) They both follow the joint-commitment before knowledge principle, where the hash commitment only needs two properties—hiding and binding; (2) they both have the strict order of message exchanges in each round; (3) they both use a digest function with the key-based uniformity and no uniform compensation properties (defined in Section 3.3); (4) they both have bound the message m_i to the commitment, and digest for SAS involves all protocol transcript.

[6] For an MP-MAP to make sense, there must be at least two non-compromised devices.

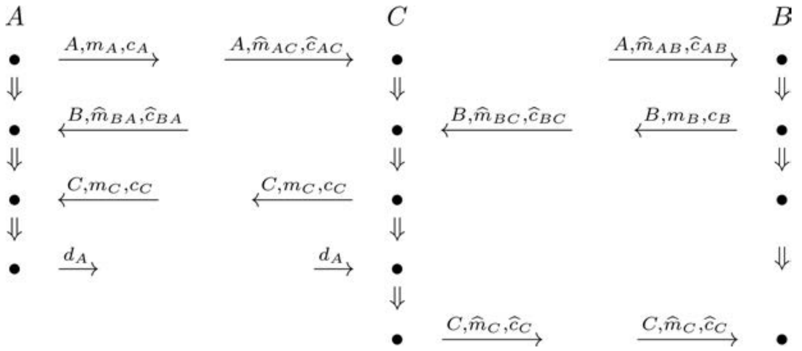

A
C
B
A,m_A,c_A
A,$\widehat{m}_{AC}$,$\widehat{c}_{AC}$
A,$\widehat{m}_{AB}$,$\widehat{c}_{AB}$
B,$\widehat{m}_{BA}$,$\widehat{c}_{BA}$
B,$\widehat{m}_{BC}$,$\widehat{c}_{BC}$
B,m_B,c_B
C,m_C,c_C
C,m_C,c_C
d_A
d_A
C,$\widehat{m}_C$,$\widehat{c}_C$
C,$\widehat{m}_C$,$\widehat{c}_C$

Fig. 8. A potential attack scenario against a three-party-MAP if the controller (C) is compromised.

5.2.1. Security Intuition of the Role of Member Count. The member count information plays an important role in achieving exclusiveness (or demonstrative identification), that is, the group authenticated in the end includes only the devices the user sees in front of her, which excludes any outsider attacker. If there is no member count information, exclusiveness cannot be achieved, as is the case in Laur and Pasini [2008], due to the fact that before the group of devices meets with each other, they do not know the member list in advance. An attacker $\mathcal{A}$ can thus claim it is one of the group members and inject her DH public key share, trying to obtain the group key. Then the actual group becomes $\widehat{\mathcal{G}} = \mathcal{G} \cup \mathcal{A}$, while for members in $\mathcal{G}$, they still have the same SAS values. While the only sign that the user perceives is the LED blinking patterns on the sensor nodes, she will accept $\widehat{\mathcal{G}}$ as authenticated. However, with the count information, this attack can be defeated. First, if $N + 1$ key shares are received by the controller, GDP will abort, assuming that the user counts correctly. Second, if M_N only receives N X_is and Y_is from $\mathcal{G}$, but $\mathcal{G} \setminus M_N$ all receive $N + 1$ key shares from $\mathcal{G} \cup \mathcal{A}$, $\mathcal{A}$ will not be able to derive the same key with all $j \in \mathcal{G}$, thus having no gain. Even if $\mathcal{A}$ carries out such an attack to disrupt the group, it will not be able to make all the SASes equal due to the properties of the digest function.

5.2.2. Security Intuition Against Compromised Devices. Here we provide more insight into why GDP is secure against compromised devices. We illustrate it using a potential attack reminiscente of the one discovered in Perković et al. [2011], that of if the controller (device with the largest ID) is compromised.

Suppose there are three devices, A, B, and C. Controller C is under the full control of the adversary, that is, it can launch active attacks in the wireless channel. Depicted in Figure 8, C tries to impersonate B to A and vice versa, but it does not try to break the group exclusiveness. C's goal is to make all the SASes equal. In the first move, after seeing c_A, C constructs new commitments $\widehat{c}_{AC}$ and $\widehat{c}_{AB}$ with $\widehat{r}_{AC}, \widehat{r}_{AB}$ known by itself. Then after B sends c_B, C does the similar thing to the preceding. In the third move, C sends C, m_C, c_C only to A to trick A into sending its decommitment d_A so that C will know r_A before this round ends. At this point, C knows all the random nonces received/generated at A and also all the received/generated data at A which leads to the revealing of sas_A in advance. What remains for C is to compute $\widehat{c}_C$ and θ offline (after seeing c_B), such that $\widehat{c}_C$ opens to an $\widehat{r}_C = r_B \oplus \theta$, where r_B is not known by C, which makes $\mathsf{sas}_B = \mathsf{sas}_A$. Assuming this can be done (since our hash commitment does not preclude malleability), C can make all SASes equal while deceiving both A and B.

In the precding attack, the attacker knows the last digest key r_N. However, if the controller is not compromised but the attacker compromises any other device with smaller ID (e.g., B), there is no way for it to obtain the value of r_N before the commitment round ends (except by breaking the hiding property with negligible probability). So there is no way to know the SAS of the controller beforehand, which also means it cannot compute the SAS of other devices (e.g., A) offline to make SASes equal.

Therefore, the key factor for MP-MAP to be immune from insider attacks is that the uncompromised controller is mandated to be the device with the largest ID. Note that in Perković et al. [2011], a similar problem is dealt with by adding another round between the commitment and decommitment rounds. Our scheme keeps the number of rounds to the minimum.

5.3. Security of Key Management

5.3.1. Secrecy of the Key Polynomial. This is ensured to be unconditionally secure and resists up to t colluding attackers [Blundo et al. 1993]. If more than t polynomial shares are collected, $f(x, y)$ can be reconstructed using bivariate Lagrange interpolation. Therefore, we set t as the maximum number of nodes in the BAN. For example, $t = 50$ is usually enough. In this case, even if all the sensors are compromised, $f(x, y)$ is secure, and we can replace compromised nodes with new ones, as long as the total number of nodes is smaller than t.

5.3.2. Backward Secrecy. For a new group member v joined during the ith session, the new group key sent out by the controller is $K_G(i+1)$. It is infeasible for v to derive $K_G(i)$, since it requires breaking the pre-image resistance property of the hash function.

5.3.3. Forward Secrecy. For a revoked former group member v, since the new group key $K_G(i + 1)$ is randomly generated by the controller and is securely delivered to the remaining group members, v can only randomly guess the value of $K_G(i + 1)$.

5.3.4. Key Update and Revocation. A revoked group member must not be able to communicate with existing members. Because the value $\Delta(i + 1)$ is randomly chosen from F_p and is encrypted thus is not known to revoked member, v can only guess it randomly. The success probability is $1/p$. For v, without knowing $\Delta(i + 1)$, even if it possesses $f_{v,i}(y)$, it cannot derive $f_{v,i+1}(y)$, therefore cannot obtain pairwise keys with any legitimate node.

6. EVALUATION

In this section, we analyze the efficiency of our device pairing and key management protocols. We first compare the overheads with an existing scheme and then report our implementation of GDP and experimental results.

6.1. Computation and Communication Efficiency of GDP

It is important for the trust establishment in a BAN to have both low computation and communication costs. A common reason is to keep low energy consumption for resource-constrained sensor devices. But more importantly, performing complex computations would increase the protocol runtime dramatically, which is not tolerable for medical monitoring applications, especially under emergency situations. Many existing group message authentication (GMA) protocols [Vaudenay 2005; Laur and Nyberg 2006; Laur and Pasini 2008, 2009; Perković et al. 2011] require the adoption of a non-malleable commitment scheme, which is usually constructed based on number-theoretic assumptions and incurs intensive computation[7] [MacKenzie and Yang 2004; Vaudenay 2005;

[7] Construction based on the hash function has also been proposed Laur and Nyberg [2006], but the security only remains as conjecture.

Table II. Comparison of MP-MAP and SAS-GMA in Terms of Overall Communication and Computation

	Decomposition	SAS-GMA (bits)	MP-MAP (bits)
Commu. cost	ID	$N \cdot \lvert ID \rvert$	$2N \cdot \lvert ID \rvert$
	commit	$N \cdot c_1 \cdot q$	$N \cdot n$
	decom.	$N \cdot (c_2 \cdot q + n + \lvert ID \rvert)$	$2N \cdot n$
	message	$N \cdot \lvert m \rvert$	$N \cdot \lvert m \rvert$
Comput. cost	hash $H(\cdot)$	$N^2 \cdot n \cdot (\lvert ID \rvert + \lvert m \rvert)$	$N^2 \cdot n \cdot (\lvert ID \rvert + \lvert m \rvert + n)$
	commit/decom.	$N \cdot (c'_1 + c'_2)\cdot$ mod_exp	$N \cdot n \cdot (\lvert ID \rvert + \lvert m \rvert + n)$
	sas	universal hash	digest function

Note: N: number of devices; q: length of group element in a non-malleable commitment scheme.

Laur and Nyberg 2006]. A representative scheme of this kind is the SAS-GMA protocol proposed in Laur and Pasini [2008, 2009], which we will compare with. In terms of computation, the biggest advantage of our MP-MAP is the elimination of non-malleable commitment schemes. Instead, we only require commitments with the basic hiding and binding properties, whereas much more efficient schemes based on hash functions can be used (while still enjoying provable security).

Therefore, we compare both the overall computation and communication overhead between our MP-MAP and the SAS-GMA in Table II. The communication overhead is evaluated in terms of the number of bits transmitted/received. For the SAS-GMA protocol, we assume the use of a non-malleable commitment scheme from MacKenzie and Yang [2004]. The constants c_1 and c_2 stand for the number of group elements (the length of each of them, q is usually 1024 bits) in the commitment and decommitment, respectively. For example, for the DSA-based commitment scheme [MacKenzie and Yang 2004], $c_1 = 2$ and $c_2 = 1$. In contrast, in the MP-MAP, we use hash commitments, and thus the length of a commitment value is the hash length, n. For instance, in SHA-256, $n = 256$, and this is much smaller than q.

For the computation overhead, the main parts come from commitment/decommitment, hash function, and SAS computation. Common to both protocols is the use of a cryptographic hash function $H(\cdot)$ to hash an arbitrary long data $(\widehat{\mathcal{G}}|\widehat{\mathbf{c}}|\widehat{\mathbf{m}})$ to the length accepted by a universal hash (e.g., 256 bits) or digest function. The complexity for a cryptographic hash is based on the simple model in Nguyen and Roscoe [2011], which is linear to both the input length and the output (or key) length. The N^2 factor is due to there being N devices, and each device's hash input length is linear with N. For the commitment/decommitment, c'_1, c'_2 refer to the number of modular exponentiations required in their computations, respectively. For the DSA-based commitment scheme [MacKenzie and Yang 2004], $c'_1 = 5, c'_2 = 4$. For the SAS, the complexity of the digest function is even smaller than a cryptographic hash [Nguyen and Roscoe 2011] and is similar to a universal hash [Laur and Pasini 2008]. In summary, it can be seen that the MP-MAP is more efficient than SAS-GMA in terms of both computation and communication.

Finally, for our GDP protocol, the additional computation overhead to the MP-MAP is also small. It requires each sensor device to perform three modular exponentiations and $2N - 2$ modular multiplications for running the UDB key agreement protocol and only one additional modular exponentiation for computing the individual key shared with the controller. The computations for setting up the pairwise keys during the deployment phase rely on Galois field multiplications instead and are much more efficient. On the other hand, the controller, which is usually more powerful, needs to carry out $N + 2$ modular exponentiations.

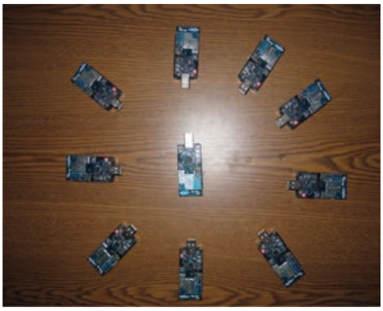


Fig. 9. Experimental setup with ten devices. The central node is designated as the controller. All nodes are displaying synchronous LED blinking patterns.

6.2. Prototype Implementation

We implemented GDP on a prototype sensor network platform consisting of ten Tmote-Sky nodes, each with an 8 MHz TI-MSP430 microcontroller, 10 KB RAM and 48 KB Flash (ROM), and TinyOS. We let one of the sensor nodes be the controller, which does not improve the performance of the GDP protocol. For our experiments, we implement rounds 2–5 in Figure 5 up to the computation of the group key and the individual keys. The counting step is omitted by programming the IDs of the devices and the group size into them in advance.

We convert the Diffie-Hellman-based group key agreement (UDB) to its elliptic curve cryptography (ECC) version, where the modular exponentiation and modular multiplication correspond to point multiplication and point addition, respectively. We use the primitive operations provided by TinyECC [Liu and Ning 2008], including point multiplication and point addition, with all optimizations enabled. To provide 80-bit key security, the finite field size used in ECC should be 160 bits. So we first compute a 160-bit group key and individual keys using ECC versions of the UDB and Diffie-Hellman key agreement and then hash the keys. In Liu and Ning [2008], for 160-bit ECC and with all optimizations enabled, the ECDH initialization time is reported to be 1.8s on Micaz, while the key computation time is 2.1s. The required ROM and RAM sizes are 16KB and 1.8KB, which are well below the capacities of a Tmote-Sky node. Since there are only four point multiplications in the ECC version of the GDP protocol on sensor nodes, GDP is fairly practical for implementation on low-end sensors.

For the hash commitment in GDP, we use a keyed hash (standard HMAC construction based on SHA-256), where the random nonce r is used as the key, and $ID|m$ is the input data. For implementation of the digest function, since the software code for it is not available, we also employ the keyed hash instead, which is only for demonstration purposes.[8] We chunk the first ℓ bits of the keyed hash to be the SAS. Finally, we set $\ell = 16$.

In Figure 9, the experimental setup is depicted. Now we describe the protocol process and user experience in more detail. After all devices are powered on, all the devices display red LED by default. Then the user presses a reset button on the controller which broadcast a reset signal to all the others. After resetting, the user presses another button on the controller to initiate the protocol. The controller's last message in each round serves as a synchronization signal, and different rounds are started/finished through state transitions on each device. In each round before the final one, the other sensors should display the same LED light pattern, which indicates that they are

[8] This only increases the computation time, since the digest function is more efficient than a hash [Nguyen and Roscoe 2008].

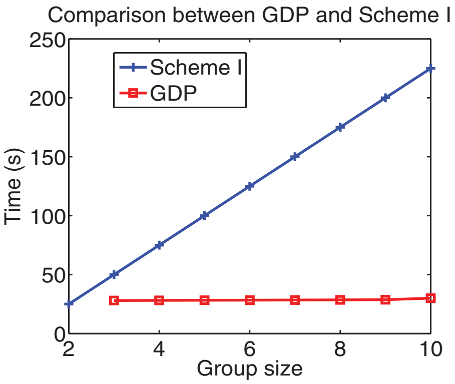


Fig. 10. Time for initial trust establishment.

Table III. Decomposition of Overhead of Each Sensor Device in GDP ($N = 10$)

Decomposition	Commu.	Comput.	LED blink.	Idle	Total
Time (ms)	409	11,005	15,360	3,187	29,961
Energy (mJ)	24.5	59.4	1,152	1.5	1,237.4

synchronized. Before devices start to display SASes, they display a green light for several seconds. The simultaneous LED blinking for SAS lasts for about 16 seconds; after that, if the patterns are the same, the user presses a button on every device to confirm. Note that in our implementation, the synchronization signals sent out by the controller are quite reliable, since the sensor nodes are put close to each other, which leads to very good channel conditions.

6.3. Results

In the following, we assume that $N_{max} = 10$, and we will show that for $N_{max} = 10$, it is practical for a human user to perform the initial trust establishment with little effort. For larger N_{max}, a specialized device could be used to aid the process, such as the one in Perković et al. [2011].

6.3.1. Time Required for Initial Trust Establishment. In our experiments, $N \leq N_{max}$. So we plot the time for one GDP run ($T_{gdp}(N)$) against the group size N in Figure 10. It can be seen that T_{gdp} is almost constant (increases linearly but very slowly) when N increases. This is because all nodes display LED blinking patterns simultaneously, while the computations are quite fast. T_{gdp} consists of time spent in computation (T_{cp}), communication (T_{cm}), and human interaction (T_I). We then decompose T_{gdp} in Table III. For $\ell = 16$ bits, $T_I \approx 16$s (one bit for 1s). Obviously, the LED blinking time takes a major portion, and then the computation time, and finally the communications. The idle time is needed for nodes to wait to receive all other's broadcasts in each round and to resolve collisions.

When $N > N_{max}$, the number of subgroups $k = \lceil \frac{N-1}{N_{max}-1} \rceil$. Then the total initial trust establishment time is

$$T_{gdp}(N) \approx (k-1)T_{gdp}(N_{max}) + T_{gdp}(N - k(N_{max} - 1)), \qquad (3)$$

which increases linearly with k and repeats the almost constant pattern when $N \leq N_{max}$. The preceding time can be approximated theoretically, based on the experimental values $T_{gdp}(N)$, $N \leq N_{max}$. For $N = 20$, $T_{gdp} \approx 60$s.

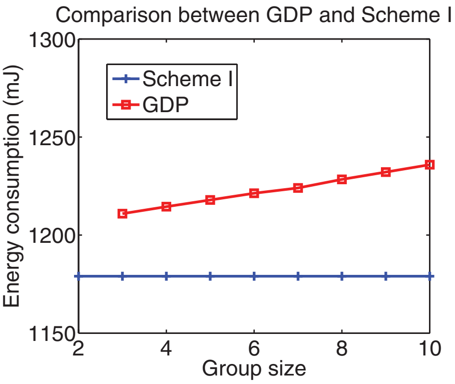


Fig. 11. Energy consumption per sensor node.

We also compare GDP with Scheme I, in which $T_{sc1}(N) = (N - 1)T_{sc1}$ (2), where $T_{sc1}(2)$ is the estimated time for pairwise device pairing. From Figure 10, $T_{sc1}(N)$ is linear with N. For $N = 20$, this is 475 s. Obviously, when $N \geq 3$, the time of GDP is far less than Scheme I, which is also the case for Keoh et al. [2009], which uses one-by-one sensor association.

6.3.2. Energy Consumption. From the data sheet of Tmo [2005], we obtain the normal voltage and current of the mote under different conditions, based on which we compute the energy consumption (EC). We plot the average EC for each sensor node in GDP against the group size ($N \leq 10$) in Figure 11, and compare it to the estimated EC of Scheme I (based on the EC breakdown for each primitive operation). The EC of GDP is a little higher than that of Scheme I, since it uses extra ECC point multiplication and addition operations. However, the difference is small (below 50 mJ). Note that for the controller, the EC of Scheme I is linear to N, which is much larger than that of GDP due to GDP's grouping mechanism.

Then we break down the EC of GDP in Table III. It can be seen that the LED blinking takes a major part in the EC, since its time is the longest and the required power is among the largest. Although the communication needs the largest power, it consumes the smallest energy, since its time is quite small. Finally, note that the energy spent in computation is very small, too, because the required power is small.

6.3.3. Usability and Security. GDP supports batch deployment. From the experiments, we found it is practical for a human to watch $n \leq 10$ LED blinking patterns simultaneously when the nodes are put close to each other. The watch-and-compare is easy to follow, and differences can be identified with high probability. While MiB [Kuo et al. 2007] and KALwEN [Law et al. 2010] also achieve batch deployment, they require additional hardware (a faraday cage (FC), a keying device, and a keying beacon). These devices add cost to the BAN and an FC is cumbersome for the user to carry. The SAS-GAKA [Laur and Pasini 2008] does not use an additional device; however, string comparison needs a user to remember strings which require N interactions. The results are summarized in Table IV. We also compare with SPATE [Lin et al. 2009], a group message authentication protocol. It requires N comparisons of T-flags for each user, while each comparison needs a few seconds, and the devices need to have a screen/display.

Finally, from the security point of view, few of the compared protocols have formal security proofs. The SAS-GAKA is proven secure under a simulation-based security model, but it requires the use of non-malleable commitment schemes. The protocol in Keoh et al. [2009] was proven secure using the Burrows-Abadi-Needham (BAN) logic,

Table IV. Comparison of GDP with Related Previous Schemes

	Comparison criteria	GDP	MiB	KALwEN	Keoh *et.al.*	SAS-GAKA	SPATE
Security	Key secrecy, authenticity	√	√	√	√	√	√
	Key confirmation	√	√	×	√	×	×
	Exclusiveness	√	√	√	√	×	√
	Provable security	√	×	×	√	√	×
Usability	Fast batch deployment	√	√	√	×	√	√
	Error-proof	√	√	√	√	×	√
	# of human interactions	$k \ll N$	/	/	N	N	N
	Human effort	L	M	M	H	M	M
Cost	Requires NO PKI	√	√	√	×	√	√
	No additional hardware	√	×	×	√	√	√
	No interface on sensors	√	√	√	√	×	×
	Involvement of PKC	L	NA	NA	M	H	L

Note: L: low; NA: none; M: medium; H: high.

but the BAN logic is mainly suitable for proving traditional authentication protocols secure, which involves the existence of pre-shared secret keys between the parties.

6.4. Efficiency of Key Management after Initial Trust Establishment

6.4.1. Communication. The overhead for adding N nodes is essentially the same as initial sensor association. The existing nodes do not need to perform extra communications. Revoking one node in subgroup k requires $\#G + |\mathcal{G}(k)| - 1$ unicasts of the controller, where $\#G$ is the number of subgroups. Our scheme is very efficient under group node leave, where the leaving nodes all belong to one subgroup or one cluster. If m nodes leave in $\mathcal{G}(k)$, the controller only needs to send $\#G + |\mathcal{G}_k| - m$ messages. Clearly, if $|\mathcal{G}_k| = N_{max}$, for single sensor leave/revocation, there is an optimal value for N_{max} which equals $\sqrt{N-1}$. For $N \leq 100$, this is smaller than 10. Therefore, it provides a guideline for choosing N_{max} for GDP.

6.4.2. Storage. If all the pairwise keys are stored along with the polynomial share, the size of the keys stored on each sensor node is $2\kappa + (N - 1 + t) \cdot \log p + n$ bits, where κ is the bit length of the symmetric key. If the sensors do not store the pairwise keys, then the minimum size of the keys is $2\kappa + t \cdot \log p + n$ bits. Assume $\kappa = 80, \log p = 80, t = 50, n = 256$, the maximum size is $4416 + 79N$ bits, while the minimum is 4416 bits. These numbers are well below 4 KByte, the available RAM of Micaz.

7. CONCLUSION

In this article, we address the problem of secure ad hoc initial trust establishment and key management in body area networks. We exploit the concept of device pairing and propose group device pairing (GDP), a novel solution that establishes an authenticated group consisting of low-end sensor devices and a controller, without relying on any predistributed secret information. An authenticated group key and individual keys are agreed upon using GDP, with the help of simultaneous and manual comparison of LED blinking patterns on all devices, which can be done within 30 seconds with enough security strength in practical applications. GDP helps the user of a BAN to visually make sure that the authenticated group only consists of those nodes that she wants to deploy and associate with the intended patient. The resulting initial key materials enable efficient key management after network deployment. We have proven the security of the proposed GDP and its two-party version (PDP) under standard security notions; especially, we show the non-necessity of non-malleable commitment schemes. Efficiency analysis shows that GDP outperforms a previous group message authentication protocol, while experimental results show that GDP greatly reduces the total time and complexity of human effort and is efficient in both communication and computation.

APPENDIX

A. SECURITY DEFINITION OF MAPS: MATCHING CONVERSATIONS

In this section, we give a formal treatment of matching conversations [Bellare and Rogaway 1994], adapt it to group settings, and deal with broadcast messages. First, if each participant $i \in \mathcal{G}$ has executed a local run (or partial run) $\mathcal{R}_i$, then we can interleave the events of all the local runs, arranging them in a single sequence, in many different ways. One of these sequences is a *topological sort* of $\{\mathcal{R}_i\}_{i\in\mathcal{G}}$ if, for all i, it preserves the order of events lying on the same $\mathcal{R}_i$. We use topological sorts to represent the notion of a proper matching up of transmission and reception events. When a protocol uses no broadcast but only point-to-point messages, we can require that we can always place a matching transmission-reception pair next to each other. We will give the definitions first for the case without broadcast and then loosen them for the case using broadcast, as is needed for our protocols.

Thus, we will say that the parties $i \in \mathcal{G}$ have *jointly matching, broadcast-free conversations* in a family $\{\mathcal{R}_i\}_{i\in\mathcal{G}}$ of local runs if there is a topological sort of the transmission and reception events of all local runs $\mathcal{R}_i$, respecting the local ordering of each $\mathcal{R}_i$, such that the following hold.

(1) Every reception event e_1 immediately follows a transmission event e_0, and e_1 receives the same message sent at e_0.
(2) Vice versa, every transmission event e_0 immediately precedes a reception event e_1, and e_1 receives the same message sent at e_0.

Thus, if the parties have matching conversations, all messages transmitted by them will be received unaltered, that is, authentically. This condition also implies that the same transmitted message is not delivered more than once, since only one reception can follow it immediately.

To generalize this notion to a group $\mathcal{G}$ with an uncorrupted subset $\mathcal{N} \subseteq \mathcal{G}$, we will suppose that associated with every reception e_1 along a local run $\mathcal{R}_i$ with $i \in \mathcal{N}$, there is an *expected sender* $j \in \mathcal{G}$. Likewise, associated with every transmission e_0 along a local run $\mathcal{R}_i$ with $i \in \mathcal{N}$, there is an *expected recipient* $j \in \mathcal{G}$. This is certainly the case with our protocols when the group $\mathcal{G}$ is known. Now, a set $\{\mathcal{R}_i\}_{i\in\mathcal{N}}$ of local runs for $i \in \mathcal{N}$ consists of jointly matching, broadcast-free matching conversations for the uncorrupted participants if there is a topological sort of the transmission and reception events of the local runs $\mathcal{R}_i$ respecting the local ordering of each $\mathcal{R}_i$ such that the following hold.

(1) For every reception event e_1, if the expected sender of e_1 is some $j \in \mathcal{N}$, then e_1 immediately follows a transmission event e_0 on $\mathcal{R}_j$, and e_1 receives the same message sent at e_0.
(2) Vice versa, for every transmission event e_0, if the expected recipient of e_0 is some $j \in \mathcal{N}$, then e_0 immediately precedes a reception event e_1 on $\mathcal{R}_j$, and e_1 receives the same message sent at e_0.

Our previous definition without corruption is equivalent to the case in which $\mathcal{N} = \mathcal{G}$, at least when the group is known, and each message makes its expected sender and expected recipient explicit.

To adapt our definition to the case with broadcast messages, we use a symbol $*$ to represent the expected recipient of a broadcast message. We assume $* \notin \mathcal{G}$. The idea is that a message with expected recipient $*$ is broadcast and may be received by everyone. In this case, there may be several reception events, all following immediately after the transmission as a block. We assume here that $\mathcal{N}$ is non-empty. A set $\{\mathcal{R}_i\}_{i\in\mathcal{N}}$ of local runs for $i \in \mathcal{N}$ consists of jointly matching conversations for the uncorrupted participants if there is a topological sort of the transmission and reception events of the local runs $\mathcal{R}_i$ respecting the local ordering of each $\mathcal{R}_i$ such that the following hold.

(1) For every transmission event e_0, e_0 immediately precedes a reception event e_1 on some $\mathcal{R}_j$, where the expected recipient of e_0 is either j or $*$. If the expected recipient of e_0 is j, then e_1 is not followed by another reception event. Moreover, e_1 receives the same message sent at e_0.

(2) For every reception event e_1, if the expected sender of e_1 is some $j \in \mathcal{N}$, then e_1 immediately follows some event e_0, and e_0 involving the same message as e_0. If e_0 is a transmission event, then e_0 lies on $\mathcal{R}_j$.

B. PROOF OF THEOREM 1

PROOF. Let the parties involved in a protocol run as A and B. The first part of the security goal is obvious, so we only need to show that for any $T_b + 2T_h + O(1)$-time adversary $\mathcal{A}$, whenever the assumptions of the theorem hold, its deception probability $\text{Adv}_\Pi(\mathcal{A})$ is no larger than $max\{\epsilon_u, \epsilon_r, \epsilon_h^2\} + max\{\epsilon_r, \epsilon_b\}$. We first denote the event "$\mathcal{A}$ succeeds in deception" as S, where

$$S = \{S_1 \wedge S_2\} \triangleq \{\mathsf{Both\text{-}accept} \wedge \mathsf{No\text{-}matching}^{\mathcal{A}}\}, \tag{4}$$

where $\mathsf{No\text{-}matching}^{\mathcal{A}}$ refers to the event that A and B do no have matching conversations. Note that in order for both of them to accept, they need to successfully verify the SASes are equal (Figure 3), and they must not abort during the protocol. It is easy to see that $\text{Adv}_\Pi(\mathcal{A}) = \Pr[S]$.

Next we analyze $\Pr[S]$. First we define view_i as the ordered set consisting of all the messages received by device i in the round 2 ($\mathsf{view}_A = \{m_A, \widehat{m}_B, c_A, \widehat{c}_B\}$, and $\mathsf{view}_B = \{\widehat{m}_A, m_B, \widehat{c}_A, c_B\}$).

We will use the following lemma to continue our proof.

LEMMA 1. *In the TP-MAP, if event S_2 happens ($\mathsf{No\text{-}matching}^{\mathcal{A}}$), then either $\mathsf{view}_A \neq \mathsf{view}_B$, or otherwise, A and B will accept with probability ϵ_b.*

The preceding is straightforward to prove. To see that, notice if $\mathsf{view}_A = \mathsf{view}_B$, in order to create no-matching conversations, the adversary must break the binding property of hash commitments (i.e., to find a different d for the same m and c values), and the probability of success is no larger than ϵ_b. Thus, we can define an event $E \triangleq \{\mathsf{view}_A \neq \mathsf{view}_B\}$.

Observe that by the total probability principle, we have the following.

$$\begin{aligned}\Pr[S] &= \Pr[S|E]P[E] + \Pr[S|\bar{E}]P[\bar{E}] \\ &\overset{(1)}{\leq} \Pr[S|E] + \Pr[S|\bar{E}] \\ &\overset{(2)}{=} \Pr[S_1|E] + \epsilon_b,\end{aligned} \tag{5}$$

where Equation (1) follows from $\Pr[E], \Pr[\bar{E}] \leq 1$, and Equation (2) follows from Lemma 1 and the fact that E implies S_2 (no-matching conversations).

Therefore, next we focus on the case that event E happens and assume $\mathcal{A}$ does not break the binding property of hash commitments. There are two cases for $\mathsf{view}_A \neq \mathsf{view}_B$

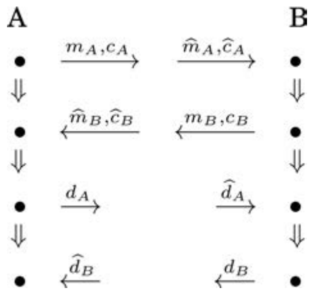


Fig. 12. Diagram for a partial execution of the protocol in the TP-MAP for our PDP. Note that here, m_i contains ID_i and the message to be authenticated.

that deserves discussion. (1) $\widehat{\mathbf{m}}_A \neq \widehat{\mathbf{m}}_B$ but $\widehat{\mathbf{c}}_A = \widehat{\mathbf{c}}_B$. This again corresponds to a double opening of the hash commitment, and the probability that the adversary will succeed in this way is bounded by ϵ_b. (2) $\widehat{\mathbf{c}}_A \neq \widehat{\mathbf{c}}_B$. We have the following lemma for this case.

LEMMA 2. *In the TP-MAP, given $\widehat{\mathbf{c}}_A \neq \widehat{\mathbf{c}}_B$, for any $T_h + O(1)$ time adversary that does not break the binding of hash commitments, $Pr[S_1|E] \leq max\{\epsilon_u, \epsilon_r\} + \epsilon_h$.*

PROOF. To be clear, consider the diagram for a partial execution (first four moves) of the protocol[9] in Figure 12. The black dots stand for the decision points of each party's run (also called a strand), while down arrows represent parties' internal state transitions. The blank parts between two strands indicate that a party's sent messages can be manipulated by any outsider adversary before they are received by the other party. The first two moves consist the first round, and the second and fourth moves consist the second round.

First let us assume the adversary $\mathcal{A}$ does not break the hiding property of hash commitments in the first round (this strategy is denote as H). This does not preclude the following three general strategies: (a) $\mathcal{A}$ can simply relay a message truthfully; (b) $\mathcal{A}$ can create a new $\widehat{c}_A$ or $\widehat{c}_B$ using $\widehat{r}_A$ or $\widehat{r}_B$ values of her own choice, but are independent of r_A and r_B; (c) create "related" $\widehat{c}_A$ and $\widehat{c}_B$ committing to *unknown* $\widehat{r}_A$ and $\widehat{r}_B$, that are correlated to r_A and/or r_B, respectively, after seeing c_A and c_B (malleability) (although the latter two are not known). The *correlation* ($\sim$) between those $\mathbf{r}$ variables could mean anything except their independence. But here, it must have a constraint—the variables (regarded as bit strings) have the same length, otherwise it does not make sense. So relations like string concatenations are excluded. The simplest relation is equality; however, relaying is the same with strategy (a), while the replay attack (copying c_A as $\widehat{c}_B$) is prevented, since the commitments have included sender ID in it, and the replay will not pass the verification of A.

Since $\mathcal{A}$ does not break the binding property of hash commitments, the digest keys are bound to the commitments, so we can focus on the commitment round only. In order to succeed, $\mathcal{A}$ must create $\widehat{c}_A$ and $\widehat{c}_B$ such that $\mathsf{sas}_A = \mathsf{sas}_B$. For $\widehat{c}_A$, if $\mathcal{A}$ chooses strategy (a) and let $\widehat{c}_A = c_A$, then r_A and r_B are independent since they are randomly generated by A and B, respectively; if $\mathcal{A}$ chooses strategy (b) or (c), due to the precedence $\widehat{c}_A \prec c_B$, $\widehat{r}_A$ will still be independent from r_B, which is unknown by $\mathcal{A}$.

As $\widehat{c}_B$ is the last message $\mathcal{A}$ can send, it must obtain a corresponding $\widehat{r}_B$ such that $\mathsf{sas}_A = \mathsf{sas}_B$. Note that $\mathcal{A}$ cannot simply relay both c_A and c_B. Next we discuss the case when $\widehat{c}_A \neq c_A$.

[9] Here we adopt the protocol representation in strand spaces proposed by Guttman [2011].

—$\mathcal{A}$ can choose strategy (b) to construct any $\widehat{c}_A$. No matter how $\widehat{c}_B$ is constructed, $\widehat{r}_A$ must be independent from both r_A and r_B, which are unknown to $\mathcal{A}$. So according to the key-based uniformity property of digest function, the probability of finding $\widehat{r}_A$ such that $\mathcal{H}(\widehat{\mathbf{m}}_A, r_A \oplus \widehat{r}_B) = y = \mathcal{H}(\widehat{\mathbf{m}}_B, \widehat{r}_A \oplus r_B)$ is smaller or equal to ϵ_u, where y is a fixed number.

—So $\mathcal{A}$ can choose strategy (a) or (c) for $\widehat{c}_A$ and $\widehat{c}_B$. According to our definitions, we have $\widehat{r}_A \sim r_A$ and $\widehat{r}_B \sim r_B$ or $\widehat{r}_B \sim r_A$. Without loss of generality, suppose $\widehat{r}_A = r_A \oplus \theta_1$ and $\widehat{r}_B = r_B \oplus \theta_2$ or $\widehat{r}_B = r_A \oplus \theta_3$. In the first case, we have $\Pr[\mathcal{H}(\widehat{\mathbf{m}}_A, r_A \oplus r_B \oplus \theta_2) = \mathcal{H}(\widehat{\mathbf{m}}_B, r_A \oplus r_B \oplus \theta_1)] \leq \epsilon_r$ according to the no uniform compensation property of digest function, where $\theta = \theta_1 \oplus \theta_2$. In the second case, we have $\Pr[\mathcal{H}(\widehat{\mathbf{m}}_A, \theta_3) = \mathcal{H}(\widehat{\mathbf{m}}_B, r_A \oplus r_B \oplus \theta_1) = y] \leq \epsilon_u$, since y is a fixed (unknown) number.

For the case $\widehat{c}_A = c_A$, it can be shown similarly that $\mathcal{A}$'s probability to succeed is no larger than $max\{\epsilon_u, \epsilon_r\}$. Combining this we get $\Pr[S_1|E] \leq max\{\epsilon_u, \epsilon_r\}$ for any $O(1)$-time adversary that does not break hiding of hash commitments.

Second, if the hiding property of any hash commitment is broken, $\mathcal{A}$'s probability of success is bounded by ϵ_h for any T_h time $\mathcal{A}$. So the Lemma is proved. □

Using Equation. (5) and Lemma 2, we get

$$\Pr[S] \leq max\{\epsilon_u, \epsilon_r\} + \epsilon_h + 2\epsilon_b, \tag{6}$$

for any adversary $\mathcal{A}$ that runs in $2T_b + T_h + O(1)$ time.

C. PROOF OF THEOREM 2

Proof. First we define view$_i$ as the set of information exchanged in the second round, which is the ordered set consisting of all the messages $(\widehat{\mathbf{m}}_i, \widehat{\mathbf{c}}_i)$, that is, $(\{\widehat{ID}_j|\widehat{X}_j|\widehat{Y}_j\}, \{\widehat{c}_j\}), j \in \mathcal{G}_i$ received by device i in round 2.

In this proof, we use $\mathcal{G}$ to denote the true group of legitimate devices (perceived by the human user) and $\mathcal{N}$ to denote the subset of non-compromised devices in $\mathcal{G}$. Similar to TP-MAP, we define

$$S = \{S_1 \wedge S_2\} \triangleq \{\text{All-accept} \wedge \text{No-matching}^{\mathcal{A}}\}, \tag{7}$$

where All-accept means that all devices in $\mathcal{N}$ accept, while No-matching$^{\mathcal{A}}$ refers to the event that there exists $i, j \in \mathcal{N}$ such that their conversations do not match. We will use the following lemma to continue our proof.

Lemma 3. *If event S_2 happens (No-matching$^{\mathcal{A}}$), then either $\exists i, j \in \mathcal{N}$ such that view$_i \neq$ view$_j$, or otherwise, all the devices in $\mathcal{N}$ will accept with probability ϵ_b.*

The argument for this lemma is similar to that of Lemma. 1.

Define event $E \triangleq \{\exists i, j \in \mathcal{N}, s.t.\text{view}_i \neq \text{view}_j\}$, we have the following.

$$\begin{aligned}
\Pr[S] &= \Pr[S|E]P[E] + \Pr[S|\bar{E}]P[\bar{E}] \\
&\overset{(1)}{\leq} \Pr[S|E] + \Pr[S|\bar{E}] \\
&\overset{(2)}{=} \Pr[S_1|E] + \epsilon_b.
\end{aligned} \tag{8}$$

The second equation follows from Lemma 3 and the fact that E implies S_2 (no-matching conversations).

Next, we use the following bound to constrain our discussion to the scenario that all pairs of non-compromised devices' SASes match, except one pair N and i (event denoted as S_{1-Ni}), $\forall i \in \mathcal{N}\backslash N$, while N and i's views do not equal. Applying the probability product rule, we get

$$\begin{aligned}\Pr[S_1|E] &= \frac{\Pr[S_1, E]}{\Pr[E]} \\ &\leq \frac{\Pr[S_1, E]}{\Pr[E, S_{1-Ni}]} \\ &= \Pr[S_{Ni}|E, S_{1-Ni}],\end{aligned} \tag{9}$$

where $S_{Ni} \triangleq \{\mathsf{sas}_N = \mathsf{sas}_i\}$, because $S_1 = \{S_{Ni}, S_{1-Ni}\}$. Also note that event E implies there must exist some i such that $\mathsf{view}_i \neq \mathsf{view}_N$, where N is the controller; we can further decompose $\Pr[S_{Ni}|E, S_{1-Ni}]$ into two cases for i and N, that is, $|\mathcal{G}_i| = N$ or $|\mathcal{G}_i| \neq N$ (note that $N = |\mathcal{G}_N|$, otherwise M_N will not accept).

Connecting the preceding we thus have the following bound on $\Pr[S]$.

$$\Pr[S_1|E] \leq max \begin{cases}\Pr[S_{Ni}|E, S_{1-Ni}, |\mathcal{G}_i| = N], \\ \Pr[S_{Ni}|E, S_{1-Ni}, |\mathcal{G}_i| \neq N].\end{cases} \tag{10}$$

It remains to show that the probabilities on the right-hand side are upper bounded by $max\{\epsilon_u, \epsilon_r\} + \epsilon_h + \epsilon_b$. We first focus on the case of $E, S_{1-Ni}, |\mathcal{G}_i| = N$.

There are two cases for $\mathsf{view}_i \neq \mathsf{view}_N$ that deserves discussion. (1) $\widehat{\mathbf{m}}_i \neq \widehat{\mathbf{m}}_N$ but $\widehat{\mathbf{c}}_i = \widehat{\mathbf{c}}_N$. This corresponds to a double opening of the hash commitment, and the probability that the adversary will succeed in this way is bounded by ϵ_b. (2) $\widehat{\mathbf{c}}_i \neq \widehat{\mathbf{c}}_N$. Here we need to consider two cases: $\mathcal{N} = \mathcal{G}$ (no compromised insiders) or $\mathcal{N} \subsetneq \mathcal{G}$ (some devices are compromised). We first discuss the former case. We have the following lemma, whose proof is shown later.

LEMMA 4. *In the MP-MAP, given $\widehat{\mathbf{c}}_i \neq \widehat{\mathbf{c}}_N$, for any T_h time adversary that does not break the binding of hash commitments, $Pr[S_{Ni}|E, S_{1-Ni}, |\mathcal{G}_i| = N] \leq max\{\epsilon_u, \epsilon_r\} + \epsilon_h$.*

For the case of $E, S_{1-Ni}, |\mathcal{G}_i| \neq N$, using a similar analysis to the proof of Lemma 4, the same conclusion can be drawn. Note that since $|\mathcal{G}_i| \neq |\mathcal{G}_N|$, in the SASes of i and N, respectively, their data input parts of the digest function will never equal each other, even if $\widehat{\mathbf{c}}_i = \widehat{\mathbf{c}}_N$ and $\widehat{\mathbf{m}}_i = \widehat{\mathbf{m}}_N$, while this does not affect the result. In fact, this is why we should include all the protocol transcript into the SAS digest.

From the preceding, we know that the right-hand side of Equation (10) is bounded by $max\{\epsilon_u, \epsilon_r\} + \epsilon_h + \epsilon_b$ for a $T_b + T_h$ time adversary. Summing up Equations (8), (9), and (10), we get $\Pr[S] \leq max\{\epsilon_u, \epsilon_r\} + \epsilon_h + 2\epsilon_b$ for a $2T_b + T_h$ time adversary. □

PROOF (LEMMA 4). Consider the simplified diagram in Figure 13. When $\mathcal{N} = \mathcal{G}$, our proof strategy is to show that if $\mathcal{A}$ does not break the hiding of any hash commitments, its probability of success will be bounded by $max\{\epsilon_u, \epsilon_r\}$. On the other hand, if any hash commitment's hiding is broken, $\mathcal{A}$'s probability of success is bounded by ϵ_h for any T_h time $\mathcal{A}$.

Then we focus on proving the first preceding statement. Adversary $\mathcal{A}$ can generate commitments $\widehat{c}_{1i}, \ldots, \widehat{c}_{i-1i}, \widehat{c}_{i+1i}, \ldots, \widehat{c}_{Ni}$ and $\widehat{c}_{1N}, \ldots, \widehat{c}_{N-1N}$ in arbitrary ways. It can either simply relay the original commitments sent by honest parties (without knowing the underlying r values) or construct new commitments using its own $\widehat{r}$ values or create commitment $\widehat{c}_{j'i'}$ that is related to any c_j, $j \in \mathcal{G}$ while not knowing $\widehat{r}_{j'i'}$, where either $j = j'$ or $j \prec j'$.

Since the last message $\mathcal{A}$ can inject/modify is $\widehat{c}_{Ni}$, we can focus on how $\mathcal{A}$ can compute it to make $\mathsf{sas}_i = \mathsf{sas}_N$. The SASes are in the following forms: $\mathsf{sas}_i = \mathcal{H}(\eta_i, \widehat{r}_{1i} \oplus \cdots \oplus r_i \oplus \cdots \oplus \widehat{r}_{Ni})$ (denoting η_i as the data inputs); and similarly, $\mathsf{sas}_N = \mathcal{H}(\eta_N, \widehat{r}_{1N} \oplus \cdots \oplus \widehat{r}_{N-1N} \oplus r_N)$. In the preceding both r_i and r_N are unknown to $\mathcal{A}$, and since $\widehat{\mathbf{c}}_i \neq \widehat{\mathbf{c}}_N$, $\eta_i \neq \eta_N$.

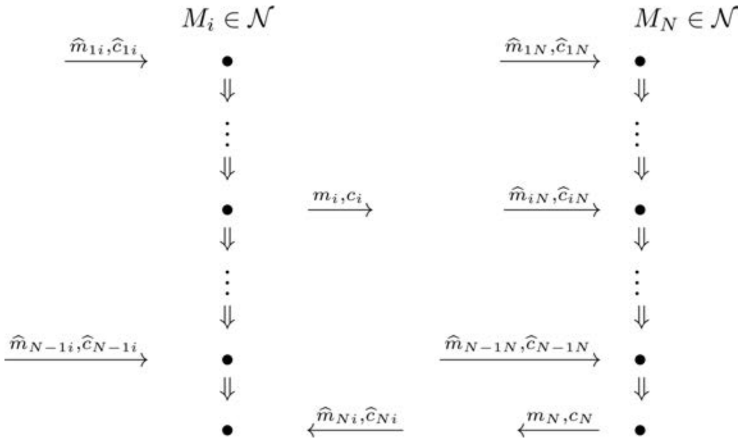

$M_i \in \mathcal{N}$
$M_N \in \mathcal{N}$
$\widehat{m}_{1i},\widehat{c}_{1i}$
$\widehat{m}_{1N},\widehat{c}_{1N}$
m_i,c_i
$\widehat{m}_{iN},\widehat{c}_{iN}$
$\widehat{m}_{N-1i},\widehat{c}_{N-1i}$
$\widehat{m}_{N-1N},\widehat{c}_{N-1N}$
$\widehat{m}_{Ni},\widehat{c}_{Ni}$
m_N,c_N

Fig. 13. Simplified diagram for a partial execution concerning devices i and N in round 2 of protocol MP-MAP. m stands for the message to be authenticated.

Note that all the $\widehat{c}_{1N},\ldots,\widehat{c}_{N-1N}$ must be created before c_N is sent out due to the message ordering, so $\widehat{r}_{1N},\ldots,\widehat{r}_{N-1N}$ must be independent of r_N, which is unknown to $\mathcal{A}$. We have the two following cases.

—If $\mathcal{A}$ relays c_N to i and $\widehat{r}_{Ni}=r_N$, since $\widehat{c}_{1i},\ldots,\widehat{c}_{i-1i},\widehat{c}_{i+1i},\ldots,\widehat{c}_{N-1i} \prec \widehat{c}_{Ni}, \widehat{r}_{1i},\ldots,\widehat{r}_{N-1i}$ must be all independent with r_N. Thus, $\mathsf{sas}_i = \mathcal{H}(\eta_i,\theta_i\oplus r_N)$, where $\theta_i=\widehat{r}_{1i}\oplus\cdots\oplus r_i\oplus\cdots\oplus\widehat{r}_{N-1i}$ is independent of r_N, and $\mathsf{sas}_N=\mathcal{H}(\eta_N,\theta_N\oplus r_N)$, where $\theta_N=\widehat{r}_{1N}\oplus\cdots\oplus r_N\oplus\cdots\oplus\widehat{r}_{N-1N}$ is independent of r_N. In the preceding no matter whether θ_i is known to $\mathcal{A}$ or not, it is a fixed number when $\mathcal{A}$ relays r_N to i, and the same is true for θ_N. In addition, $\eta_i\neq\eta_N$. So according to the no uniform compensation property of digest function, $\Pr[\mathcal{H}(\eta_i,\theta_i\oplus r_N)=\mathcal{H}(\eta_N,\theta_N\oplus r_N)]\leq\epsilon_r$, and $\theta=\theta_i\oplus\theta_N$.

—If $\widehat{c}_{Ni}$ is created by $\mathcal{A}$ using other strategies. Because $\mathcal{A}$ is free to create related commitments to c_i after seeing c_i and is also free to create its own commitments, it could make $\theta_i\oplus\widehat{r}_{Ni}$ equal to a number θ_i' it knows (otherwise, there will be an unknown factor r_j in $\widehat{r}_{1i}\oplus\cdots\oplus\widehat{r}_{Ni}$, which reduces to the same case as preceding). Also, sas_N is fixed when $\mathcal{A}$ sends $\widehat{r}_{Ni}$ to i. So the key-based uniformity property of digest function applies, and $\Pr[\mathcal{H}(\eta_i,\theta_i')=\mathcal{H}(\eta_N,\theta_N\oplus r_N)]\leq\epsilon_u$.

Finally, for the scenario with compromised devices ($\mathcal{N}\subsetneq\mathcal{G}$), the only additional information to $\mathcal{A}$ is the internal r_j values for $j\in\mathcal{G}\setminus\mathcal{N}, j\neq i, j\neq N$. It is easy to see that the preceding proof still holds as long as i and N are not compromised (r_i, r_N are not known by $\mathcal{A}$). □

ACKNOWLEDGMENTS

The authors would like to thank Hanfei Zhao for his help with GDP's prototype implementation. We also thank Shahab Mirzadeh for his comments on the conference version of this article. Finally, we thank the anonymous reviewers for their helpful comments.

REFERENCES

Alliance, W. 2006. Association models supplement to the certified wireless universal serial bus specification. *Revision 1*, 3.

Ateniese, G., Steiner, M., and Tsudik, G. 2000. New multiparty authentication services and key agreement protocols. *IEEE J. Select. Areas Commun. 18,* 4, 628–639.

BALFANZ, D., SMETTERS, D. K., STEWART, P., AND WONG, H. C. 2002. Talking to strangers: Authentication in ad-hoc wireless networks. In *Proceedings of the Network and Distributed System Security Symposinm (NDSS02)*.

BELLARE, M., CANETTI, R., AND KRAWCZYK, H. 1998. A modular approach to the design and analysis of authentication and key exchange protocols (extended abstract). In *Proceedings of the 13th Annual ACM Symposium on Theory of Computing*. 419–428.

BELLARE, M. AND ROGAWAY, P. 1994. Entity authentication and key distribution. In *Proceedings of the 13th Annual International Crypotology Conference on Advance in Cryptology*, vol. 773. Springer-Verlag, Berlin, 232–249.

BLUNDO, C., SANTIS, A. D., HERZBERG, A., KUTTEN, S., VACCARO, U., AND YUNG, M. 1993. Perfectly-secure key distribution for dynamic conferences. In *Proceedings of the 12th Annual International Crypotology Conference on Advance in Cryptology (CRYPTO'92)*. Lecture Notes in Computer Science, vol. 740, Springer-Verlag, Berline, 471–486.

CAGALJ, M., CAPKUN, S., AND HUBAUX, J.-P. 2006. Key agreement in peer-to-peer wireless networks. *Proc. IEEE 94,* 2, 467–478.

CHAN, H., PERRIG, A., AND SONG, D. 2003. Random key predistribution schemes for sensor networks. In *Proceedings of the Symposium on Security and Privacy.* 197.

CHEN, C.-H. O., CHEN, C.-W., KUO, C., LAI, Y.-H., MCCUNE, J. M., STUDER, A., PERRIG, A., YANG, B.-Y., AND WU, T.-C. 2008. Gangs: Gather, authenticate 'n group securely. In *Proceedings of the International Conference on Mobile Emputing and Networking (MobiCom'08)*. 92–103.

DI PIETRO, R., MANCINI, L., AND MEI, A. 2003. Random key-assignment for secure wireless sensor networks. In *Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks*. 62–71.

DU, W., DENG, J., HAN, Y., VARSHNEY, P., KATZ, J., AND KHALILI, A. 2005. A pairwise key predistribution scheme for wireless sensor networks. *ACM Trans. Information Syst. Security (TISSEC) 8,* 2, 228–258.

DUTTA, R. AND BARUA, R. 2008. Provably secure constant round contributory group key agreement in dynamic setting. *IEEE Trans. Inf. Theory 54,* 5, 2007–2025.

ESCHENAUER, L. AND GLIGOR, V. D. 2002. A key-management scheme for distributed sensor networks. In *Proceedings of the Conference on Computer and Communications Security (CCS'02)*. 41–47.

GOODRICH, M. T., SIRIVIANOS, M., SOLIS, J., TSUDIK, G., AND UZUN, E. 2006. Loud and clear: Human-verifiable authentication based on audio. In *Proceedings of the IEEE International Conference on Distributed Computer Systems.*

GUTTMAN, J. 2011. Shapes: Surveying crypto protocol runs. In *Formal Models and Techniques for Analyzing Security Protocols, Cryptology and Information Security Series. IOS Press*.

HALEVI, S. AND MICALI, S. 1996. Practical and provably-secure commitment schemes from collision-free hashing. In *Proceedings of the 16th Annual International Crypotology Conference on Advance in Cryptology.* Lecture Notes in Computer Science, vol. 1109, Springer, Bertin, 201–215.

HANSON, M., POWELL, H., BARTH, A., RINGGENBERG, K., CALHOUN, B., AYLOR, J., AND LACH, J. 2009. Body area sensor networks: Challenges and opportunities. *Computer 42,* 1, 58–65.

JANA, S., PREMNATH, S., CLARK, M., KASERA, S., PATWARI, N., AND KRISHNAMURTHY, S. 2009. On the effectiveness of secret key extraction from wireless signal strength in real environments. In *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking*. 321–332.

JOVANOV, E., MILENKOVIC, A., OTTO, C., AND DE GROEN, P. C. 2005. A wireless body area network of intelligent motion sensors for computer assisted physical rehabilitation. *J. Neuroeng. Rehabil. 2,* 1.

KEOH, S. L., LUPU, E., AND SLOMAN, M. 2009. Securing body sensor networks: Sensor association and key management. In *Proceedings of the IEEE International Conference on Pervesive Computing and Communication (PerCom'09)*, 1–6.

KUMAR, A., SAXENA, N., TSUDIK, G., AND UZUN, E. 2009. Caveat eptor: A comparative study of secure device pairing methods. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communication (PerCom'09)*, 1–10.

KUO, C., LUK, M., NEGI, R., AND PERRIG, A. 2007. Message-in-a-bottle: User-friendly and secure key deployment for sensor nodes. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (SenSys'07)*. 233–246.

LAMPORT, L. 1981. Password authentication with insecure communication. *Commun. ACM 24,* 11, 770–772.

LAUR, S., ASOKAN, N., AND NYBERG, K. 2005. Efficient mutual data authentication using manually authenticated strings. In *Proceedings of the International Conference on Cryptology and Network Security*. Lecture Notes in Computer Science, vol. 4301, Springer, Berlin, 90–107.

LAUR, S. AND NYBERG, K. 2006. Efficient mutual data authentication using manually authenticated strings. In *Proceedings of the International Conference on Cryptology and Network Security*. Lecture Notes in Computer Science, vol. 4301, Springer, Berlin, 90–107.

LAUR, S. AND PASINI, S. 2008. SAS-Based Group Authentication and Key Agreement Protocols. In *Proceedings of the International Conference on Public Key Cryptography (PKC'08)*. Lecture Notes in Computer Science, vol. 4939, Springer-Verlag, Berlin, 197–213.

LAUR, S. AND PASINI, S. 2009. User-aided data authentication. *Int. J. Secur. Netw. 4,* 1, 69–86.

LAW, Y., MONIAVA, G., GONG, Z., HARTEL, P., AND PALANISWAMI, M. 2010. Kalwen: A new practical and interoperable key management scheme for body sensor networks. *Secur. Commun. Netw. 4*, 11, 1309–1329.

LI, M., LOU, W., AND REN, K. 2010a. Data security and privacy in wireless body area networks. *IEEE Wirel. Commun. 17,* 1, 51–58.

LI, M., LOU, W., AND REN, K. 2010b. Secure device pairing. In *Encyclopedia of Cryptography and Security* 2nd Ed, Springer, Berlin.

LI, M., YU, S., LOU, W., AND REN, K. 2010. Group device pairing based secure sensor association and key management for body area networks. In *Proceedings of the Joint Conference of the IEEE Computer and Communication Societies.* 1–9.

LIN, Y.-H., STUDER, A., HSIAO, H.-C., MCCUNE, J. M., WANG, K.-H., KROHN, M., LIN, P.-L., PERRIG, A., SUN, H.-M., AND YANG, B.-Y. 2009. Spate: Small-group pki-less authenticated trust establishment. In *Proceedings of the ACM International Conference on Mobile System, Applications, and Services (MobiSys'09)*. 1–14.

LIU, A. AND NING, P. 2008. Tinyecc: A configurable library for elliptic curve cryptography in wireless sensor networks. In *Proceedings of the International Conference on Information Processing in Senser Networks (IPSN'08)*. 245–256.

LIU, D. AND NING, P. 2003. Establishing pairwise keys in distributed sensor networks. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS'03)*. 52–61.

LIU, D., NING, P., AND DU, W. 2008. Group-based key predistribution for wireless sensor networks. *ACM Trans. Sen. Netw. 4,* 2, 1–30.

LORINCZ, K., MALAN, D., FULFORD-JONES, T., NAWOJ, A., CLAVEL, A., SHNAYDER, V., MAINLAND, G., WELSH, M., AND MOULTON, S. 2004. Sensor networks for emergency response: Challenges and Opportunities. *IEEE Pervasive Comput. 3,* 4, 16–23.

MACKENZIE, P. AND YANG, K. 2004. On simulation-sound trapdoor commitments. In *Proceedings of the International Conference on the Theory and Applications of Cryptographic Technigues*. Lecture Notes in Computer Science, vol. 3072. Springer, Berlin, 382–400.

MALAN, D., WELSH, M., AND SMITH, M. 2004. A public-key infrastructure for key distribution in tinyos based on elliptic curve cryptography. In *Proceedings of the IEEE Internatinal Conference on Sensor and Ad Hoc Communication and Networks*. 71–80.

MALASRI, K. AND WANG, L. 2007. Addressing security in medical sensor networks. In *Proceedings of the 1st International Workshop on Systems and Networking Support for Healthcare and Assisted Living Environment (HealthNet'07)*. 7–12.

MATHUR, S., TRAPPE, W., MANDAYAM, N., YE, C., AND REZNIK, A. 2008. Radio-telepathy: Extracting a secret key from an unauthenticated wireless channel. In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*. 128–139.

MCCUNE, J. M., PERRIG, A., AND REITER, M. K. 2005. Seeing-is-believing: Using camera phones for human-verifiable authentication. In *Proceedings of the IEEE Sympesiem on Security and Privacy.* 110–124.

MORCHON, O., BALDUS, H., AND SANCHEZ, D. 2006. Resource-efficient security for medical body sensor networks. In *Proceedings of the International Conference on Wearable and Implantable Body Sensor Networks (BSN'06)*. 83.

NGUYEN, L. AND ROSCOE, A. 2008. Authenticating ad hoc networks by comparison of short digests. *Inform. Computa. 206,* 2–4, 250–271.

NGUYEN, L. AND ROSCOE, A. 2011. Authentication protocols based on low-bandwidth unspoofable channels: a comparative survey. *J. Comput. Secur. 19,* 1, 139–201.

NITHYANAND, R., SAXENA, N., TSUDIK, G., AND UZUN, E. 2010. Groupthink: Usability of secure group association for wireless devices. In *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*. 331–340.

PASINI, S. AND VAUDENAY, S. 2006. SAS-based authenticated key agreement. In *Proceedings of the 9th International Conference on Theory and Practice of Public Key Cryptography (PKC'06)*. Lecture Notes in Computer Science, vol. 3958. Springer, Berlin, 395–409.

PASS, R. 2003. On deniability in the common reference string and random oracle model. In *Proceedings of the Annual International Cryptology Conference on Advances in Cryptology.* Lecture Notes in Computer Science, vol. 2729, Springer, Berlin, 316–337.

PERKOVIĆ, T., ČAGALJ, M., MASTELIĆ, T., SAXENA, N., AND BEGUŠIĆ, D. 2011. Secure initialization of multiple constrained wireless devices for an unaided user. *IEEE Trans. Mobile Comput. 11*, 2, 337–351.

PERRIG, A., SZEWCZYK, R., TYGAR, J., WEN, V., AND CULLER, D. 2002. Spins: Security protocols for sensor networks. *Wirel. Netw. 8,* 5, 521–534.

POON, C., ZHANG, Y.-T., AND BAO, S.-D. 2006. A novel biometrics method to secure wireless body area sensor networks for telemedicine and m-health. *IEEE Commun. Mag. 44,* 4, 73–81.

PRASAD, R. AND SAXENA, N. 2008. Efficient device pairing using human-comparable synchronized audiovisual patterns. In *Proceedings of the International Conference on Applied Cryptography and Network Security (ACNS).* Lecture Notes in Computer Science, vol. 5037, 328–345.

SINGH, K. AND MUTHUKKUMARASAMY, V. 2007. Authenticated key establishment protocols for a home health care system. In *Proceedings of the International Conference on Series on Intelligent Sensors, Sensors Networks and Information Processing (ISSNIP'07)*. 353–358.

STAJANO, F. AND ANDERSON, R. J. 2000. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Proceedings of the 7th International Workshop on Security Protocols (IWSP'00)*. 172–194.

TAN, C. C., WANG, H., ZHONG, S., AND LI, Q. 2008. Body sensor network security: an identity-based cryptography approach. In *Proceedings of the 1st ACM Conference on Wireless Network Security (WiSec'08)*. 148–153.

TMOTE. 2005. Tmote-Sky product description key features. http:// www.bandwavetech.com/download/tmote-sky-datasheet.pdf.

VAN LAERHOVEN, K., SCHMIDT, A., AND GELLERSEN, H.-W. 2002. Multi-sensor context aware clothing. In *Proceedings of the 6th IEEE International Symposium on Wearable Computers (ISWC'02)*. 49–56.

VAUDENAY, S. 2005. Secure communications over insecure channels based on short authenticated strings. In *Proceedings of the Annual International Cryptology on Advances in Cryptology*. Lecture Notes in Computer Science, vol. 3621, Springer, Berlin, 309–326.

VENKATASUBRAMANIAN, K., BANERJEE, A., AND GUPTA, S. 2010. Pska: Usable and secure key agreement scheme for body area networks. *IEEE Trans. Inform. Technol. Biomed. 14,* 1, 60–68.

VENKATASUBRAMANIAN, K. AND GUPTA, S. 2010. Physiological value-based efficient usable security solutions for body sensor networks. *ACM Trans. Sen. Netw. 6,* 4, 1–36.

VENKATASUBRAMANIAN, K., GUPTA, S., JETLEY, R., AND JONES, P. 2010. Interoperable medical devices: Communication security issues. *IEEE Pulse 1,* 2, 16–27.

WONG, C. K., GOUDA, M., AND LAM, S. S. 1998. Secure group communications using key graphs. *SIGCOMM Comput. Commun. Rev. 28,* 4, 68–79.

ZHU, S., SETIA, S., AND JAJODIA, S. 2003. Leap: Efficient security mechanisms for large-scale distributed sensor networks. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS'03)*. 62–72.

ZHU, S., SETIA, S., AND JAJODIA, S. 2006. Leap+: Efficient security mechanisms for large-scale distributed sensor networks. *ACM Trans. Sen. Netw. 2,* 4, 500–528.

ZIMMERMANN, P., JOHNSTON, A., AND CALLAS, J. 2006. Zrtp: Extensions to rtp for diffie-hellman key agreement for srtp draft-zimmermann-avt-zrtp-01. http://tods.ietf.org/html/draft-zimmermann-avt-zrtp-0.

Received June 2011; revised September 2011; accepted October 2011