ReTrust: Attack-Resistant and Lightweight Trust Management for Medical Sensor Networks

Daojing He, Chun Chen, Sammy Chan, Jiajun Bu, and Athanasios V. Vasilakos

***Abstract*—Wireless medical sensor networks (MSNs) enable ubiquitous health monitoring of users during their everyday lives, at health sites, without restricting their freedom. Establishing trust among distributed network entities has been recognized as a powerful tool to improve the security and performance of distributed networks such as mobile ad hoc networks and sensor networks. However, most existing trust systems are not well suited for MSNs due to the unique operational and security requirements of MSNs. Moreover, similar to most security schemes, trust management methods themselves can be vulnerable to attacks. Unfortunately, this issue is often ignored in existing trust systems. In this paper, we identify the security and performance challenges facing a sensor network for wireless medical monitoring and suggest it should follow a two-tier architecture. Based on such an architecture, we develop an attack-resistant and lightweight trust management scheme named *ReTrust*. This paper also reports the experimental results of the Collection Tree Protocol using our proposed system in a network of TelosB motes, which show that ReTrust not only can efficiently detect malicious/faulty behaviors, but can also significantly improve the network performance in practice.**

***Index Terms*—Attack-resistance, medical sensor networks (MSNs), network performance, security, trust management.**

I. Introduction

NOWADAYS, sensor and wireless communication technologies are rapidly evolving and capturing new application areas in the healthcare domain. Medical sensor networks (MSNs) are becoming more popular and powerful, allowing for ubiquitous usage of a wide range of medical applications, such as chronic disease management. These networks allow for unobtrusive and pervasive health monitoring of the users, enhancing the quality of health monitoring systems [1] at hospital or in other healthcare facilities.

Manuscript received April 29, 2011; revised January 3, 2012; accepted April 1, 2012. Date of publication April 16, 2012; date of current version July 5, 2012. This work was supported by Scholarship Award for Excellent Doctoral Student granted by Ministry of Education, National Science Foundation of China under Grant 61070155, Program for New Century Excellent Talents in University (NCET-09-0685), and a grant from the Research Grants Council of the Hong Kong, SAR, China, under Project City U 111208.

D. He, C. Chen, and J. Bu are with the Zhejiang Provincial Key Laboratory of Service Robot, College of Computer Science, Zhejiang University, Hangzhou 310027, China (e-mail: hedaojinghit@gmail.com; chenc@cs.zju.edu.cn; bjj@zju.edu.cn).

S. Chan is with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong, SAR (e-mail: eeschan@cityu.edu.hk).

A. V. Vasilakos is with the Department of Computer and Telecommunications Engineering, University of Western Macedonia, 50100 Kozani, Greece (e-mail: vasilako@ath.forthnet.gr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITB.2012.2194788

Meeting the strict security and performance needs of these ubiquitous medical applications is a big challenge, since safety and privacy of medical data have to be guaranteed all the way from the sensor nodes (SNs) to the base station (BS), and the system has to fulfill latency needs while limited resource of SNs are expected [2]. Establishing trust among distributed network entities has been recognized as a powerful tool to improve the security and performance of distributed networks such as mobile ad hoc networks (MANETs) and wireless sensor networks (WSNs). A number of trust management protocols have been proposed for Internet security (e.g., [3] and [4]), MANETs (e.g., [5]–[7]), and WSNs (e.g., [8]–[11]). However, very few trust management schemes have been proposed for MSNs. To our knowledge, *TrE* [12] is the only trust-evaluation model related to an MSN, which is proposed for secure multicast. We observe that since all these works (e.g., [3]–[12]) do not consider the unique operational and security requirements of MSNs, they might not be suitable for MSNs. Moreover, it should be noted that similar to most security schemes, trust management methods themselves can be vulnerable to attacks [13], such as bad-mouthing attack and on–off attack. However, most existing trust evaluation mechanisms (e.g., [3]–[12]) do not take these attacks into account. Due to these reasons, there is a growing demand for adequate provision of an attack-resistant and lightweight trust management protocol for MSNs.

The major contributions of this paper are threefold.

1) We identify the security and performance challenges facing a sensor network for wireless medical monitoring. Then, we suggest a two-tier architecture for an MSN. The analysis shows that it is indispensable for increasing overall network capacity and scalability, reducing system complexity, and prolonging network lifetime.

2) Based on the proposed two-tier architecture, we develop an attack-resistant and lightweight trust management protocol named *ReTrust* which remedies the security and efficiency weaknesses of existing trust systems. ReTrust is lightweight, since it does not impose any additional overhead on the resource-poor SNs and the trust calculation on master nodes (MNs) is simple. To the best of our knowledge, this is the first attack-resistant trust management protocol for MSNs.

3) We implement the Collection Tree Protocol (CTP) using TrE and ReTrust, respectively, in a network of TelosB motes. Experimental results show that ReTrust not only effectively identifies malicious behaviors and excludes malicious/faulty nodes, but also significantly improves the network performance in practice. To our knowledge, this is also the first implementation of trust management system on the mote platform. Also, simulation

results show that ReTrust can efficiently defend on–off attack and bad-mouthing attack.

The rest of this paper is organized as follows. Section II gives an overview of related works. Section III presents an overview of MSNs and the threat model. The trust calculation of ReTrust is described in Section IV. Section V describes the system structure of ReTrust in details. Then in Section VI, security analysis, performance analysis, and functionality evaluation of ReTrust are given. Section VII concludes this paper.

II. Related Works

Trust information of individual entities among distributed networks can be used to data aggregation [8], assist routing [10], malicious node detection, and even time synchronization. Particularly, trust management has emerged as an essential complementary function to cryptographic mechanisms.

The research on trust evaluation has been extensively performed for a wide range of applications in Internet security, including E-Commerce (e.g., [3]) and peer-to-peer networks (e.g., [4]). However, empirical studies have demonstrated that traditional security strategies for wired networks do not work well in wireless networks, due to the special characteristics of wireless communications [12]. In the literature, a number of trust management protocols have been proposed for MANETs (e.g., [5]–[7]) and WSNs (e.g., [8]–[11]). TrE [12] is the only trust evaluation model related to an MSN, which is proposed for secure multicast. By simulations, the authors have shown the security and efficiency of TrE are better than currently accepted trust schemes (e.g., [5]). Also, with regard to trust management, the VITRUVIUS project [14] has explored the system properties and behavior of the application components of the body hub.

Due to the unique features and application requirements of MSNs (the detailed description will be given in Section V-A), all these works ([3]–[12]) are not suitable for MSNs. An example is that with the commonly used battery-operated and low bit-rate body SNs, only limited computation and communication capability are available. However, all these trust management protocols calculate trust in a fully distributed manner, in which each SN not only monitors the behaviors of other nodes but also manages the trust records for them. As a result, these mechanisms [3]–[12] incur high costs on SNs in terms of processing power, memory, bandwidth, and energy consumption, which do not meet the resource constraints of the SNs of an MSN.

For TrE, each node needs to preselect some threshold values (e.g., κ and the scaling factor λ of TrE) that make their scheme nonadaptive. Also, each node only relies on its direct monitoring for calculating trust value, which makes it vulnerable against collaborative attacks. Most importantly, the features and application requirements of MSNs are not identified and considered in the design of TrE. Therefore, TrE is too simple to ensure the security and efficiency of MSNs. Moreover, in the design of most existing trust systems (e.g., [3]–[12]), security vulnerabilities of a trust system (e.g., bad-mouthing attack and on–off attack) are not considered.

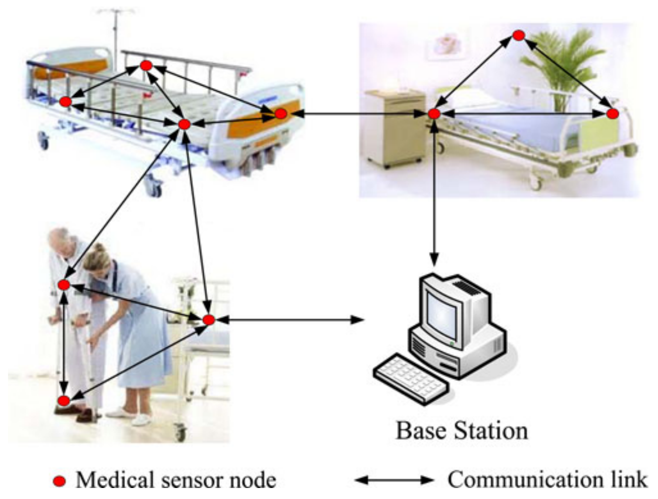


Fig. 1. Schematic diagram of an MSN.

III. Overview of MSNs and Threat Model

A. Overview of MSNs

An MSN accommodates tens or hundreds of users' body sensor networks (BSNs) [15], other SNs (e.g., sensing the temperature of a specific room) and relay nodes. Each BSN mainly consists of tiny wireless SNs that are placed in, on, or around a patient's body. These sensors consistently monitor patients' physiological activities and actions, such as health status and motion pattern. The sensed data from all BSNs may be sent to one local server for data processing, aggregation, or permanent records. Wireless sensors could replace existing wired telemetry systems for many specific medical applications, such as long-term ambulatory monitoring. Fig. 1 depicts an exemplary hospital MSN [16]. The emergence of low-power, single-chip radios based on the bluetooth and 802.15.4 standards has precipitated the design of small-networked medical sensors.

B. Threat Model

We assume that MSNs potentially face many threats, due to the sensitive nature of the data they collect and the broadcast nature of the wireless communication environment. The threats originate from two sources: active and passive attackers. Active attackers have the capability to drop messages, modify messages, inject forged messages, replay old messages, send a large volume of bogus packets to jam the communication channels, compromise nodes, or spoof nodes. Active attackers not only invade patients' privacy but also suppress legitimate data or insert a bogus one into the network leading to unwanted actions (e.g., drug delivery) or blocking legitimate actions (e.g., notifying doctor in case of an emergency). In this paper, the terms action and behavior are used interchangeably. On the other hand, although passive attackers do not try to interfere with the functions of the MSN, they are capable of eavesdropping on all traffic within an MSN. Also, they may potentially be able to perform offline cryptanalytical attacks to access confidential data being communicated, thereby invading patients' privacy.

Similar to most security schemes, trust management methods themselves can be vulnerable to attacks such as bad-mouthing attack, on–off attack, sybil attack and newcomer attack. In bad-mouthing attack, malicious parties may provide dishonest recommendations to frame good parties and/or boost trust values of malicious peers. On–off attack means that malicious entities behave well and badly alternatively, hoping that they can remain undetected while causing damage. In sybil attack, a malicious node creates several faked IDs for the trust management system. If a malicious node can easily register as a new user, trust management suffers from the newcomer attack in which a malicious node can easily remove its bad history by registering as a new user. The former two attacks should be considered in the design of a trust management protocol. However, the defense against the latter two attacks does not rely on the design of trust management, but on authentication and access control [17], which make registering a new or faked ID difficult.

IV. Trust Calculation of ReTrust

In this section, we present the trust calculation procedure of ReTrust in details.

A. *The Definition of Trust*

Trust is defined as a belief level that one node can put on another node for a specific action according to previous direct or indirect information from observation of behaviors [18]. The belief level is the extent to which one node believes that another node is willing and able to obey the protocol and act normally. For example, regarding forwarding packets, node x observes that node y forwards some packets for it normally and drops the remaining packets, which is specified as direct information. This can be done either through overhearing, or based on link layer acknowledgement.

The trust record stores information about trust relationships and associated trust values. A trust relationship is always established between two parties for a specific action. That is, one party trusts the other party to perform an action. In this work, the first party is referred to as the *subject* and the second party as the *agent*. The notation $\{subject{:}agent, action\}$ is used here to represent a trust relationship. For each trust relationship, a numerical value $T(subject{:}agent, action)$, referred to as *trust value*, describes the level of trustworthiness. There are two common ways to establish trust in wireless networks. First, when the subject can directly observe the agent's behavior, direct trust can be established. Second, when the subject receives recommendations from other entities about the agent, indirect trust can be established. In this paper, a trust value is considered to be an integer in $[0, \lambda]$, where 0 denotes the most untrusted state, while λ denotes the most trusted state. In our experiments, we set $\lambda = 100$.

Next, we describe the calculation of direct, recommendation, and indirect trusts.

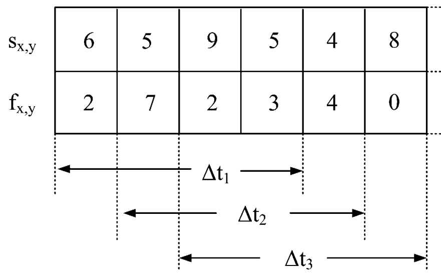


Fig. 2. Sliding time window scheme of ReTrust.

B. *Attack-Resistant Management of Direct Trust*

We assume a node x observes that the numbers of successful interactions and failed interactions of node y are s and f, respectively. According to the beta-function-based method [19], the direct trust value can be calculated as $T_{x,y} = \frac{s+1}{s+f+2}$. If $s = f = 0$, then $T_{x,y} = \frac{1}{2}$. However, we observe that the network traffic conditions (e.g., congestion and delay) should not affect the trust attached to a node. This means that the trust calculation should not emphasize the timing information of each interaction too rigidly. Additionally, a node's historical trust values should be taken into account in order to measure its current trustworthiness. To solve these issues, we use a sliding time window concept in the trust calculation as follows. The time window $\triangle T$ is used to measure the number of successful and failed interactions. It consists of several time units. The interactions that occur in each time unit within the time window are recorded. After a unit of time elapses, the window slides one time unit forward, thereby dropping the interactions done during the first unit. Thus, as time progresses, the window forgets the experiences of one unit but adds the experiences of the latest time unit. The window length could be made shorter or longer based on network analysis scenarios. A sample scenario of the ReTrust time window scheme is illustrated in Fig. 2. The time window $\triangle T$ consists of four units. During the first unit of $\triangle t_1$, the numbers of successful and failed interactions are 6 and 2, respectively. After the elapse of the first unit, the new time window $\triangle t_2$ drops the interaction values that took place during the very first unit of $\triangle t_1 (s = 6, f = 2)$ and only considers the values of the last three units of $\triangle t_1$ plus values of one recent unit added on the right $(s = 4, f = 4)$.

With this time window information, the time-based past interaction trust value $T_{x,y}$ of node y at node x that lies on $[0, \lambda]$ is defined as

$$T_{x,y} = \left\lfloor \lambda \times \frac{\sum_{j=1}^{m} \beta_j \times \frac{s_{x,y}^j + 1}{s_{x,y}^j + f_{x,y}^j + 2}}{\sum_{j=1}^{m} \beta_j} \right\rfloor \tag{1}$$

where $\lfloor u \rfloor$ returns the largest integer which is smaller than or equal to u, $s_{x,y}^j$ is the number of successful interactions of node x with y during the jth unit of $\triangle t$, and $f_{x,y}^j$ is the number of failed interactions of node x with y during the jth unit of $\triangle t$. The parameter m is the number of time units in each time window $\triangle t$. For example, $m = 4$ in Fig. 2. Here, we introduce $0 \le \beta_j \le 1$ as the aging factor, which describes that the trust value made long time ago should carry less importance than the trust value made more recently. Therefore, $\beta_1 < \beta_2 \ldots < \beta_m$. When node y's behavior changes rapidly, the observations made long time ago are not very useful for predicting node y's future behavior. In this case, β_j should increase at a faster rate with j, and vice versa. The use of the aging factor provides a way to capture dynamic changes in node y's behaviors. Here, for β_j, we use the exponential decrease method to deemphasize old observations. More specifically, $\beta_j = \varphi^{m-j}$, where $0 < \varphi < 1$.

In most trust calculation procedures [e.g., (1)], bad behavior can be compensated with good behavior, malicious nodes can take advantage of it and behave well and badly alternatively. Hence they can remain trusted while behaving badly. Thus, to resist on–off attack, the calculation of $T_{x,y}$ is modified as follows. Instead of using a fixed aging factor, we set $\beta_j = \beta_j \times (1 - p^j)$, where $p^j = \frac{s^j_{x,y}+1}{s^j_{x,y}+f^j_{x,y}+2}$. The idea of such a defense is that not only the trust value made long time ago should carry less importance than the trust value made more recently, but also bad behavior is remembered for a longer time than good behavior. As a result, $T_{x,y}$ is given by

$$T_{x,y} = \left\lfloor \lambda \times \frac{\sum_{j=1}^{m} \beta_j \times (1-p^j) \times p^j}{\sum_{j=1}^{m} \beta_j \times (1-p^j)} \right\rfloor. \tag{2}$$

C. Attack-Resistant Management of Recommendation Trust

Recommendation trust is a special type of direct trust. It is set for trust relationship $\{subject$: $agent$, making correct recommendations$\}$. There are two ways for node x to obtain the recommended trust value of node y about performing an action *act* from other nodes. One is that node x checks its trust records and then selects a set of nodes, denoted by Ψ, which have the trust value larger than a threshold. Subsequently, node x transmits a recommendation request message to Ψ through multicasting. Obviously, the identity of node y and the other related information (e.g., the action *act*) should be added into the recommendation request. Upon receiving a request message, the qualified nodes will reply if they have information needed by node x. The other approach is that node x broadcasts a recommendation request message to its neighbors and waits for replies. Node x sets the hop number h of the recommendation request message propagation and then adds h to the request message. Upon receiving a request message, the neighbors will reply if they have information needed by node x. At the same time, they will update the hop field of the request message as $h = h - 1$. If $h > 0$, they will forward the request message to their one-hop neighbors; otherwise, they will simply drop the request message.

Next, the subject judges whether a recommendation from the agent is correct or not. If node x can detect that the recommendation reported from node y_i is false, node y_i's recommendation is evaluated to be bad; otherwise, node x believes node y_i's recommendation is good. This judgment is done by outlier detection schemes (e.g., checking consistency between observations and recommendations, or among multiple recommendations). We consider a simple checking among multiple recommendations as an example as follows.

Suppose that node y_i is a one-hop neighbor of node x and node y_i has a trust value $T^p_{y_i,z}$ of node z about performing packet forwarding, node x broadcasts the recommendation request for node z in the one-hop area and then receives $T^p_{y_i,z}$, where $i \in \{1, \ldots, n\}$. After that, node x calculates the mean me and the standard deviation sd of the dataset $\{T^p_{y_1,z}, T^p_{y_2,z}, \ldots, T^p_{y_n,z}\}$. The recommendations within $[me - 2 \times sd, me + 2 \times sd]$ are evaluated to be good.

Thus, same as (2), the subject x computes the recommendation trust value of the agent y_i, where s^j_r and f^j_r are the number of good and bad recommendations received from the agent y_i during the jth unit, respectively.

As long as recommendations are taken into consideration, ReTrust employs the following two mechanisms to defend against bad-mouthing attack. On the one hand, as described earlier, recommendation trust is treated separately from regular direct trust, and can only be established based on previous recommendation behaviors. Thus, the subject can detect and isolate the malicious nodes who have provided bad recommendations. On the other hand, only good recommendations are used in the calculation of indirect trust described below.

D. Attack-Resistant Management of Indirect Trust

Indirect trust can be established between two nodes that have not previously interacted since trust is transitive. For example, node y observes the behavior of node z about performing action act and makes recommendation to node x as $T^{act}_{y,z} = T(y : z, act)$ [calculated by (2)]. Node x trusts node y with $T\{x : y$, making correct recommendations$\}(=R_{x,y})$. The calculation of the recommendation trust $R_{x,y}$ has been given in Section IV-C. The question is how much node x should trust node z to perform the action act. With the recommendation trust value $R_{x,y}$ as the weight of indirect information received $T^{\text{act}}_{y,z}$, one way to calculate $T_{x,y,z} = T(x : z, act)$ is

$$T_{x,y,z} = \left\lfloor \frac{R_{x,y} \times T^{\text{act}}_{y,z}}{\lambda} \right\rfloor. \tag{3}$$

This phenomenon is called trust propagation. Indirect trust is established through trust propagation. Many trust models have been proposed to determine how to calculate indirect trust between two nodes from trust propagation paths. To prevent bad-mouthing attack, a necessary condition to trust propagation is added into the indirect trust calculation. That is, trust can propagate along path x–y–z if the recommendation trust between node x and y is greater than a threshold. In a general multihop recommendation path, this condition is held in each intermediate node. For example, trust can propagate along path x–y–z–w if the recommendation trust between node x (respectively, y) and y (respectively, z) is greater than a threshold.

V. System Structure of ReTrust

A. Unique Features and Application Requirements of MSNs

In this section, some differences between MSNs and MANETs (respectively, WSNs) are listed as follows [20]. 1) Latency: this requirement is dictated by the applications, and may be traded for improved security and energy consumption. However, while energy conservation is always important, replacement of batteries in MSNs nodes is much easier done than that in WSNs, whose nodes can be physically unreachable after deployment. Thus, it may be necessary to maximize battery life time in a WSN at the expense of higher latency. 2) Flexibility: noninvasive sensors can be used to automatically monitor physiological readings, which can be forwarded to nearby devices (e.g., a PDA or mobile phone) according to application requirements. 3) Effectiveness and efficiency: the signals that body sensors collect can be effectively processed to obtain reliable and accurate physiological estimations. Also, their ultra-low power consumption makes their batteries long lasting.

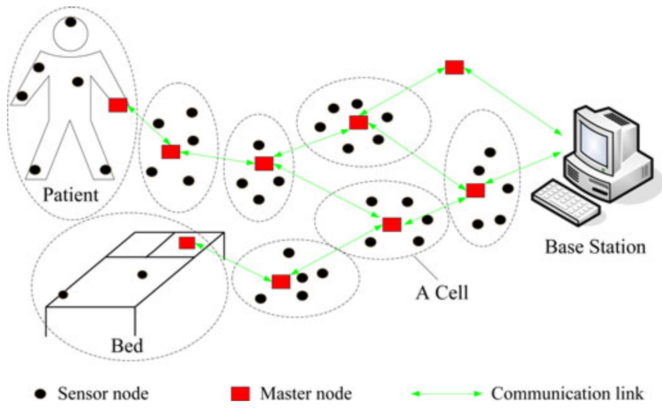


Fig. 3. Proposed two-tier architecture of an MSN.

B. Two-tier Network Architecture

Some security studies on MSNs such as MITHril [21] and SMART [22] utilize cables to directly connect multiple body sensors with an actuator node (e.g., a PDA). Alternatively, the CodeBlue project [23] stipulates that body sensors directly communicate with the BS without involving any actuator node. Also, the VITRUVIUS project [24] suggests that body sensors are connected to a hub and the hub is used as a gateway. Generally, the architecture of an MSN is considered to be a flat network as suggested earlier in [2], [12], [16] and [24]–[26].

Different from the previous works (e.g., [2], [12], [16] and [21]–[26]), we suggest an MSN should follow a two-tier architecture, which is a kind of hierarchical network. The whole network region is partitioned into a collection of cells, each containing an MN in charging of a number of SNs. SNs are mainly responsible for sensing tasks, while MNs perform more resource-demanding computation and communication tasks. We assume that MNs and SNs know their affiliated cells. Note that tiering does imply physical clustering, each SN only communicates with a sole MN. MNs and SNs differ significantly in their resources. In particular, MNs have abundant resources in storage, energy (e.g., a heavy-duty battery or solar panel), and computation. Also, each MN can communicate with neighboring MNs via relatively long-range, high-rate 802.11-like radios, thus forming an upper-tier multihop network. In contrast, SNs are much more constrained in every regard. Additionally, we assume that time is divided into epochs. At the end of each epoch, each SN submits to its affiliated MN all the data (if any) it generated during that epoch through a single-hop, low-power, low-rate, short-distance radio-link-like CC1100 or 802.15.4. For example, in an MSN, each BSN is a cell, in which the user has one MN (e.g., Imote2 node, PDA, iPhone or mobile phone) attached to his/her body or bedside. The MN is connected to several or tens of medical SNs through one-hop communication links, which take samples of the user's health data. Fig. 3 shows a two-tier architecture of an MSN.

Compared to the previous approaches (e.g., [2], [12], [16], and [21]–[26]), such a two-tier architecture is indispensable for increasing overall network capacity and scalability, reducing system complexity, prolonging network lifetime and ensuring the security and privacy. For example, tiers are fundamental to scaling the whole network size and spatial extent, since MNs collectively have greater network capacity and larger spatial reach than a flat (non-tiered) field of SNs. Also, it can achieve substantial power saving at the body sensors as they only have to transmit over a short range. Additionally, such an *ad hoc*-based architecture facilitates fast deployment when encountering a dynamic environment, such as medical emergency care response, or at a disaster site. More importantly, in a BSN, user's MN can be configured with an access policy that controls who has privilege to access the medical data within his/her BSN.

C. Overview of ReTrust

Obviously, the SNs in an MSN cannot manage the trust records of other nodes due to the limited resource. So, different from traditional trust management systems [3]–[12], ReTrust only requires the MN of each cell to manage the trust records of other MNs and all cell member SNs, that is, ReTrust works with two topologies. One is the intracell topology, where an MN manages the trust records of all the cell member SNs based on past direct interaction. The other is intercell topology, where each MN manages the trust records of other MNs based on past direct observation, recommendation, and indirect interaction.

D. Trust Management in the Intracell Level

As described in Section IV-B, in a cell, each cell member SN is within the transmission range of the MN and vice versa. Thus, in ReTrust, only direct information from observation of behaviors of each SN is employed to calculate its trust value. There are many possible actions SNs would carry out in a cell of MSNs depending on different applications. According to the features of an MSN, data processing is introduced into the trust management. The detailed description is given as follows.

The quality of the data (e.g., temperature and light) reported by an SN can be used to represent the node's behavior in data processing task. In a cell, the MN can detect whether the data reported from an SN during a time unit is false. If yes, the interaction of the node is failed; otherwise, this is a successful interaction. This detection is often done through outlier detection (e.g., checking consistency among multiple data). Obviously, through replacing multiple recommendations with multiple data, the simple checking described in Section IV-C can also be employed here.

According to the aforementioned analysis, the MN in each cell can manage the trust records of all the cell member SNs about performing multiactions. It should be noted that bad behaviors are not equally bad. For example, if an SN submits false sensing data, even one bad behavior should be sufficient to exclude this SN from the network. On the other hand, if an SN drops some packets, we are not sure whether this SN is really malicious or not. Thus, in order to allow for different application circumstances, there is an apparent necessity to weigh each action relative to the magnitude it endows on the total trust value. Here, we assume an MN A calculates the total trust value T_{AB}^{total} of its cell member SN B associated with multiactions as follows:

$$T_{AB}^{\text{total}} = \epsilon_1 \times T(A:B, act_1) + \cdots + \epsilon_p \times T(A:B, act_p) \tag{4}$$

where ϵ_i $(1 \leq i \leq p)$ are weights for each of the actions, $0 \leq \epsilon_i \leq 1$, and $\sum_{i=1}^{p} \epsilon_i = 1$. Each weight is proportional to the significance of an action to the calculation of the total trust value. The larger the weight of a specific action, the more important that action is to the total trust value and vice versa. It is suggested that the weight of each action should be carefully chosen according to the specific application scenario.

E. Trust Management in the Intercell Level

As described in Section IV-A, each MN manages the direct trust records of its one-hop neighboring MNs through observing their behaviors. At the same time, as described in Sections IV-B and C, each MN manages the recommendation and indirect trust records of its non-one-hop neighboring MNs. Also, each MN submits all these records to the BS. Upon receiving these information, the BS can run some efficient centralized mechanism to detect the malicious MNs. An example mechanism is the well-known alerts reasoning algorithm [27], which takes the form $(t, \text{MN}_1, \text{MN}_2)$, indicating that MN_1 observes an abnormal activity of MN_2 at time unit t (e.g., the trust value of MN_2 about performing an action act is lower than a threshold). This algorithm takes into account the possibility that compromised nodes may collude at will. Comprehensive experiments of [27] have shown that this algorithm is optimal in the sense that it identifies the largest number of compromised nodes without introducing false positives.

VI. Security Analysis, Efficiency Analysis, and Functionality Evaluation

A. Security and Efficiency Analysis

Recently, a number of mechanisms (e.g., [17], [28], and [29]) based on traditional cryptographic technique (e.g., symmetric cryptography, digital signature) have been proposed to protect the security of MSNs. Obviously, similar to most trust management protocols, these techniques can be directly employed in ReTrust to ensure the confidentiality, integrity, authentication, access control, and nonrepudiation. Thus, not only can ReTrust prevent the attackers from eavesdropping, dropping, modifying, injecting, and replaying messages, but also resist sybil and newcomer attacks on trust evaluation methods.

Also, ReTrust can provide many security and privacy protection. For example, within a cell, the MN will be able to manage the direct trust records of the member SNs and detect bad SNs. Also, each MN can manage the trust records of other MNs and detect bad MNs. Thus, trust management helps to select trusted en route MNs through which the sender will forward data to the BS. The prediction of nodes' future behavior directly determines the risk faced by the MSN [respectively, a specific cell (e.g., one user's BSN)]. Thus, stronger security mechanisms should be employed when risk is high. Additionally, with the assessment of trustworthiness of individual network entities, it is possible to evaluate the trustworthiness of the MSN (respectively, a specific cell).

Moreover, ReTrust is lightweight in two aspects. First, different from all existing trust management approaches (e.g., [3]–[12]), ReTrust does not impose any additional overhead on the SNs. Second, as described in Section V, the trust management just runs on each resource-rich MN, and the trust calculation is simple.

B. Malicious Node Detection

We investigate the management of trust records by simulation that reveals important insight into the effects of several attack/antiattack methods presented earlier in the text. The simulation is set up as follows. Each MN randomly selects one of its neighbors to transmit packets. Suppose that MN A asks MN B to forward packets, MN A can observe how many packets B has forwarded. Next, MN A manages its trust record T(A: B, packet forwarding) according to (2). At the same time, as described in Sections IV-C and D, MN A manages the trust records of two-hop neighbor MNs. In our experiment, a legitimate MN drops the packets from other MNs with packet drop ratio less than 10%. A malicious MN drops the packets from legitimate MNs with a packet drop ratio of 75%. Each sliding window consists of ten time units. If the trust value is less than 30, the MN is evaluated to be malicious; otherwise, the MN is evaluated to be legitimate. In this simulation, the number of forwarded packets in each time unit is set to 500.

Here, we introduce a metric MND to describe the malicious node detection performance. Let $\boldsymbol{D}$ denote the set of malicious MNs that MN A (which is randomly chosen from the legitimate MNs) has detected. $\boldsymbol{M}$ denotes the set of malicious MNs, which are MN A's neighbors. Then, MND is defined as $\frac{|\boldsymbol{D}|}{|\boldsymbol{M}|}$. Thus, MND is a real number in $[0, 1]$.

1) On–Off Attack: Here we add a pseudorandom number generator (PRNG) for each malicious MN, which randomly generates a number ranging from 0 to 1 for each time unit. For the generated random number, when it is less than 0.2, the MN behaves badly (i.e., drops the packets) in the time unit. Also, in each ten time units (i.e., each sliding window), each malicious MN behaves badly for more than two time units. For the system to reach the steady state, the simulation first runs for 200 time units and then on–off attacks are launched (the simulation then runs for 300 time units). Two systems are compared: 1) on–off

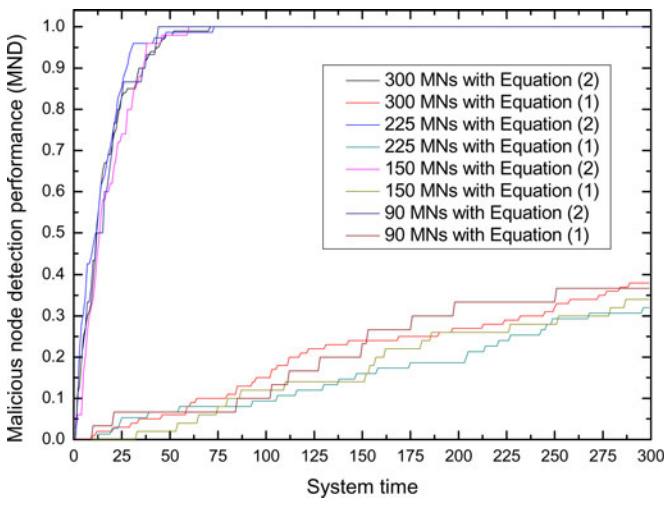


Fig. 4. Effect of the number of MNs on system stability when the system is under on–off attack (the percentage of malicious MNs is fixed at 33.3%).

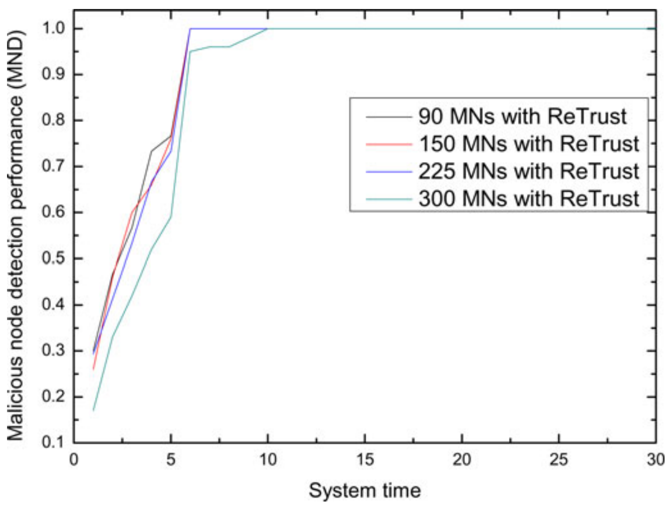


Fig. 6. Effect of the number of MNs on system stability when the system is under bad-mouthing attack (the percentage of malicious MNs is fixed at 33.3%).

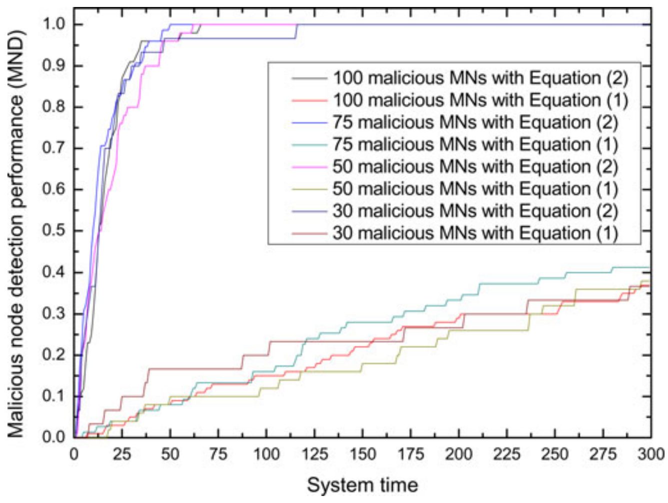


Fig. 5. Effect of the number of malicious MNs on system stability when the system is under on–off attack (the number of MNs is fixed at 300).

attack with (1) (i.e., the defense against on–off attack is not added) and 2) on–off attack with (2) (i.e., the defense against on–off attack is activated). To show the stability of the system as it scales up, we verify the MND against the number of MNs with a fixed % of malicious MNs (33.3% in our experiment). As shown in Fig. 4, compared to (1), the MND with (2) increases much more quickly and can achieve 100% after a short time (i.e., about 50 time units). With the increase of the number of MNs, the ReTrust system with (2) remains stable.

In Fig. 5, we plot the MND against the number of malicious MNs with a fixed number of MNs (i.e., 300 MNs). Similarly, compared to (1), MND with (2) increases much more quickly and can achieve 100% after a short time (i.e., 66 time units). With the increase of the number of malicious MNs, the ReTrust system with (2) remains stable.

2) Bad-Mouthing Attack: Here, we consider a smart adversary, where it does not launch bad-mouthing attack constantly, in order to protect itself from being detected. Thus, as described earlier, we also add a PRNG for each malicious MN. For the generated random number, when it is less than 0.2, the MN behaves badly (i.e., submit incorrect recommendations) in the time unit. In a bad-mouthing attack, to frame good parties, malicious MNs will reduce the trust values of honest MNs by 20 through recommendation (of course, the trust value of a honest MN should be more than 30). Also, in each sliding window, each malicious MN behaves badly for more than two time units (i.e., submits two incorrect recommendations). The adversary may launch bad-mouthing attacks from the beginning of the simulation.

Fig. 6 shows the MND against the number of malicious MNs with a fixed % of malicious MNs (33.3% in our experiment). The MND with the ReTrust system increases very quickly and can achieve 100% after a short time (i.e., 9 time units). With the increase of the number of MNs, the ReTrust system remains stable.

Fig. 7 shows the MND against the number of malicious MNs with a fixed number of MNs (i.e., 300 MNs). The MND with ReTrust system increases quickly and can achieve 100% after a short time (i.e., 21 time units). With the increase of the number of MNs, the ReTrust system remains stable.

C. *Network Throughput Improvement*

We evaluate network throughput improvement by implementing the CTP using ReTrust on an experimental test bed.

In an MSN, the collection of medical data from each SN to the BS can be done using the traditional CTP. As the reference routing protocol for TinyOS 2.x [30], CTP has been strenuously tested and shown to work well in mote networks. We use an indoor test bed consisting of 20 cells and a BS (performed by a TelosB mote) to compare the performances of traditional CTP, the CTP using TrE and the CTP using ReTrust. Each cell consists of an MN (performed by one TelosB mote) and two cell member SNs (performed by two TelosB motes). The TelosB mote has an 8-MHz CPU, 10 kB RAM, 48-kB ROM, 1 MB of flash memory, and an 802.15.4/ZigBee radio. These motes run TinyOS 2.1.0.

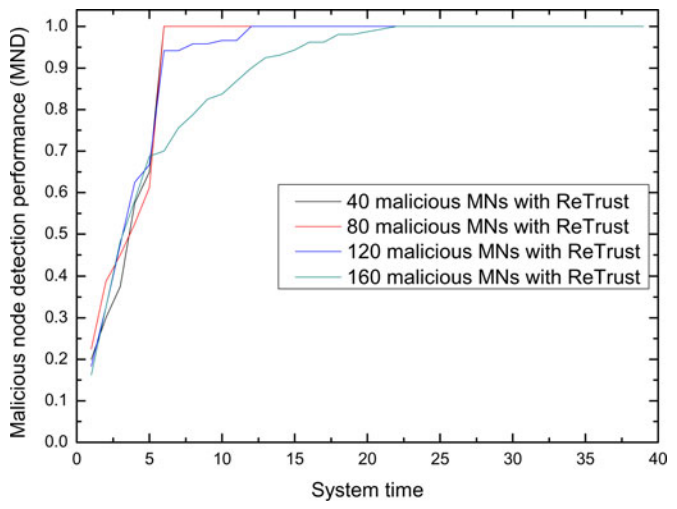


Fig. 7. Effect of the number of malicious MNs on system stability when the system is under bad-mouthing attack (the number of MNs is fixed at 300).

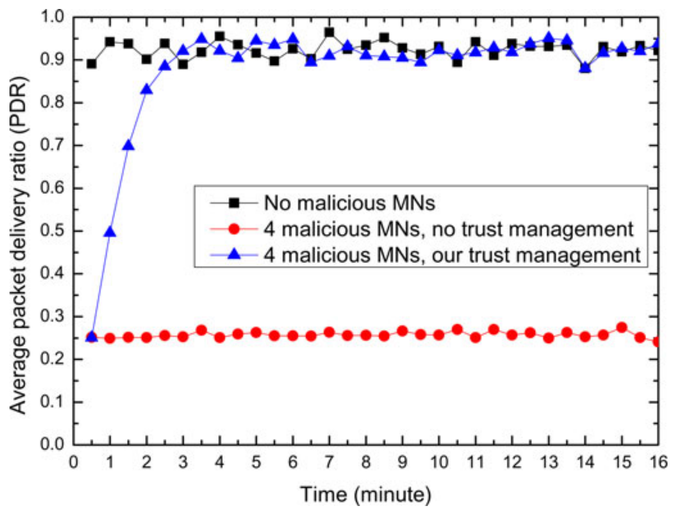


Fig. 8. Network throughput with and without ReTrust.

We deploy 20 MNs in a 4×5 grid, where the separation between neighboring grid points is about 0.5 m. Among the 20 MNs, the possible number of malicious nodes is from 0 to 4. For each SN, the delivery rate of packets is based on the random sending mechanism of CTP, which is about 0.34 packets/s. The data collecting MNs are located along one edge (width) of the grid, which are far from the BS. The BS calculates the average packet delivery ratio (PDR) every 30 s.

In this implementation, we focus on the *TestNetwork* module which employs CTP as its collection protocol. The malicious MNs perform gray-hole attack, i.e., randomly dropping 85% packets passing through them. Same as the legitimate MNs, malicious MNs will acknowledge the packets that they received. It should be noted that different from malicious MNs, congested MNs notify neighbors about network congestion. Note that traditional CTP cannot observe the packet dropping behavior of malicious nodes, although it employs the bidirectional expected transmissions (ETX) scheme. A more detailed explanation is given as follows. CTP contains three main subsystems: link estimator, routing engine, and forwarding engine [30]. Routing estimation and selection mainly depend on link estimator. Link estimator uses link qualities to evaluate the neighbors based on beacons and acknowledgements (Acks). Since a malicious node can also echo Acks as a legitimate one, this can cause its children to choose it as their parent. Additionally, traditional CTP restricts a parent for deciding whether its children are legitimate ones. Hence in traditional CTP, a malicious node can disguise as a good one, while probably causing a great amount of data missing in MSNs.

For simplicity, here, we only consider the case in which each MN manages trust records of its one-hop neighbor MNs by observing their packet-forwarding activities as described in Section V-E. The number of time units in each time window (m) is set to 10. To introduce ReTrust, we have modified the basic CTP as follows. A trust value about performing packet forwarding is attached to each member of the neighbor table of each MN. While deciding the next hop of a route, each MN, say MN_1, will choose a threshold value as the trust requirement of the next hop node of this route. MN_1 then checks the total trust values of all of its one-hop neighboring MNs and selects those neighbors which meet this trust requirement. After that, MN_1 checks the qualified MNs' ETX and the MN with the best ETX will be chosen as the next hop of this route. We modify *Routing Engine* and *Link Estimator* modules of CTP to achieve the aforementioned goals.

In the first experiment, three systems are compared: 1) traditional CTP without malicious MNs, 2) the CTP without trust management but with four malicious MNs which will be randomly chosen, 3) the CTP with four malicious MNs and ReTrust. Fig. 8 shows the percentage of successfully transmitted packets, which represents network throughput, as a function of time. Obviously, malicious MNs can significantly degrade the network throughput of traditional CTP. After employing ReTrust, the network throughput can be recovered because the CTP using ReTrust efficiently avoids choosing malicious MNs as parents to forward packets to the BS. Also, when the execution time increases, ReTrust can bring the network performance close to that with no malicious MNs very quickly because more accurate trust records are built up over time.

In the second experiment, we change the total number of malicious MNs from 0 to 4 and the malicious MNs will be randomly chosen. Three systems are compared: 1) traditional CTP; 2) the CTP using TrE; 3) the CTP using ReTrust. Fig. 9 shows the percentage of packets successfully transmitted. The experiment lasts about 1920s. Obviously, malicious MNs can significantly degrade the performance of basic CTP and the CTP using TrE. Even with two malicious MNs (10% of total MNs), the average PDR of these two mechanisms can be as low as 60%. The reasons for this are as follows. First, TrE considers successful packet forwarding but not taking failed packet forwarding into account. Second, in TrE, it is difficult for each MN to choose a proper threshold for trust calculation. Third, the packet forwarding behavior of an MN is not considered in traditional CTP. It can be seen that using ReTrust to build and utilize trust records about performing packet forwarding can greatly improve the

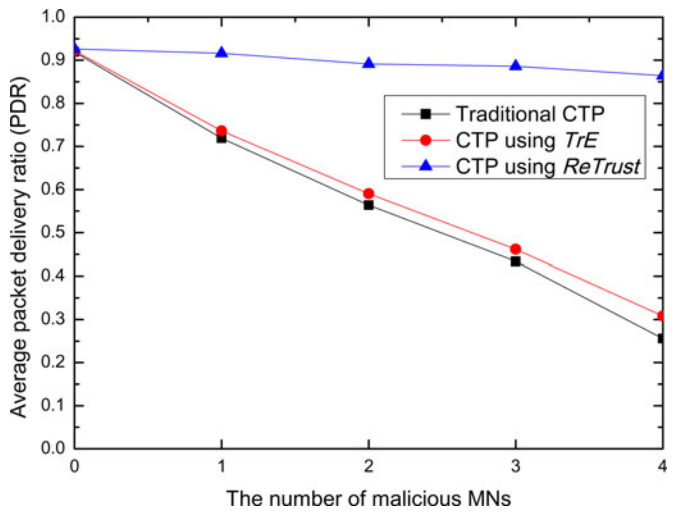


Fig. 9. Network throughput with different number of malicious MNs.

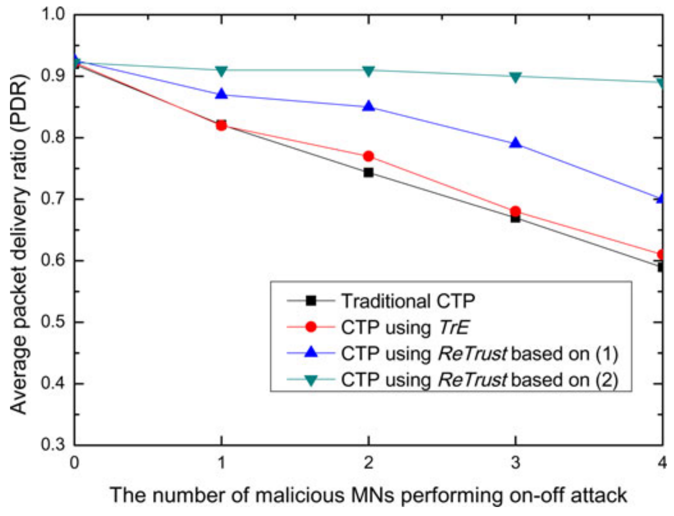


Fig. 10. Network throughput with different number of malicious MNs performing on–off attack.

performance. The main difference between these three systems is that, compared to the traditional CTP and CTP using TrE, the CTP using ReTrust efficiently avoids choosing malicious MNs as parents to forward packets to the BS. This experiment also indicates that ReTrust can efficiently detect bad nodes and then isolate them.

The robustness of ReTrust based on (2) under on–off attack is investigated in the third experiment. We change the total number of malicious MNs from 0 to 4 and the malicious MNs are randomly chosen. Different from the second experiment, here malicious MNs drop packets through performing on–off attack. Specifically, malicious MNs behave well 30 s and behaves badly 30 s alternatively. Four systems are compared: 1) traditional CTP, 2) the CTP using TrE, 3) the CTP using ReTrust based on (1), and 4) the CTP using ReTrust based on (2). The experiment also lasts about 1920s. Fig. 10 shows that with the increase of the number of malicious MNs, the network throughput of the former three mechanisms falls. When there are three malicious MNs (15% of total MNs), the average PDRs of the former two mechanisms are as low as 68%, while the average PDRs of the latter two systems are 79% and 90%, respectively. Even with four malicious MNs (20% of total MNs), the average PDR of the CTP using ReTrust based on (2) can be as high as 89%. This is because, by introducing the defense against on–off attack, the trust value keeps track of the malicious MNs' current status after the entities alter their activities. Therefore, we can see that ReTrust based on (2) can efficiently resist on–off attack.

VII. Conclusion

With the emergence of widespread use of MSNs, the need of a proper trust management protocol is strongly felt. In this paper, an attack-resistant and lightweight trust management scheme named ReTrust for MSNs has been proposed. The security analysis and experimental results have shown that ReTrust is feasible for enhancing the security and network performance of real MSN applications.

References

[1] ZigBee Alliance, "Personal, home, and hospital care: technical requirements document," *Release 075111r02*, Sep. 2007.

[2] O. G. Morchon and H. Baldus, "Efficient distributed security for wireless medical sensor networks," in *Proc. Intell. Sensors, Sensor Netw. Inf. Process. 2008*, pp., 249–254.

[3] T. Grandison and M. Sloman, "A survey of trust in internet applications," *IEEE Commun. Surveys Tuts.*, vol. 3, no. 4, pp. 2–16, Fourth Quarter 2000.

[4] D. Ingram, "An evidence based architecture for efficient, attack-resistant computational trust dissemination in peer-to-peer networks," in *Proc. iTrust 2005*, pp., 273–288.

[5] G. Theodorakopoulos and J. S. Baras, "Trust evaluation in ad-hoc networks," in *Proc. ACM Conf. Wireless Netw. Security 2005*, pp. 1–10.

[6] R. Venkataraman, M. Pushpalatha, and T. Rao, "A generalized trust framework for mobile ad hoc networks," in *Recent Trends in Networks and Communications*, vol. 90, N. Meghanathan, S. Boumerdassi, N. Chaki, D. Nagamalai, Eds. Berlin/Heidelberg, Germany: Springer-Verlag, 2010, pp. 326–335.

[7] K. Wang, M. Wu, and S. Shen, "A trust evaluation method for node cooperation in mobile ad hoc networks," in *Proc. Int. Conf. Inf. Technol. 2008*, pp., 1000–1005.

[8] S. Ganeriwal, L. K. Balzano, and M. B. Srivastava, "Reputation-based framework for high integrity sensor networks," *ACM Trans. Sensor Netw.*, vol. 4, no. 3, pp. 1–37, 2008.

[9] A. Boukerch, L. Xu, and K. EL-Khatib, "Trust-based security for wireless ad hoc and sensor networks," *Comput. Commun.*, vol. 30, no. 11–12, pp. 2413–2427, 2007.

[10] Y. Stelios, N. Papayanoulas, P. Trakadas, S. Maniatis, H. Leligou and T. Zahariadis, "A distributed energy-aware trust management system for secure routing in wireless sensor networks," in *Mobile Lightweight Wireless Systems,* vol. 13, F. Granelli, C. Skianis, Y. Xiao, and S. Redana, Eds. Berlin, Germany: Springer-Verlag, 2009, pp. 85–92.

[11] K. Liu, N. Abu-Ghazaleh, and K.-D. Kang, "Location verification and trust management for resilient geographic routing," *J. Parallel Distrib. Comput.*, vol. 67, no. 2, pp. 215–228, 2007.

[12] A. Boukerche and Y. Ren, "A secure mobile healthcare system using trust-based multicast scheme," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 4, pp. 387–399, May 2009.

[13] Y. Sun, Z. Han, and K. J. R. Liu, "Defense of trust management vulnerabilities in distributed networks," *IEEE Commun. Mag.*, vol. 46, no. 2, pp. 112–119, Feb. 2008.

[14] V. T. Bui, "A trust management model for body sensor networks," in *Proc. IEEE Int. Symp. World Wireless, Mobile Multimedia*, 2011, pp. 1–3.

[15] M. Chen, S. Gonzalez, A. Vasilakos, H. Cao, and V. Leung, "Body area networks: A survey," *ACM/Springer Mobile Netw. Appl.*, vol. 16, no. 2, pp. 171–193, 2011.

[16] K. Malasri and L Wang, "Addressing security in medical sensor networks," in *Proc. HealthNet 2007*, pp. 7–12.

[17] D. He, J. Bu, S. Zhu, S. Chan, and C. Chen, "Distributed access control with privacy support in wireless sensor networks," *IEEE Trans. Wireless Comm.*, vol. 10, no. 10, pp. 3472–3481, Oct. 2011.

[18] J. Li and J. Kato, "Future trust management framework for mobile ad hoc networks," *IEEE Commun. Mag.*, vol. 46, no. 4, pp. 108–114, Apr. 2008.

[19] A. Josang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decis. Support Syst.*, vol. 43, no. 2, pp. 618–644, 2005.

[20] A. Pentland, "Healthwear: medical technology becomes wearable," *Computer*, vol. 37, no. 5, pp. 42–49, 2004.

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

[22] D. Curtis, E. Shih, J. Waterman, J. Guttag, J. Bailey, T. Stair, R. Greenes, and L. Ohno-Machado, "Physiological signal monitoring in the waiting areas of an emergency room," in *Proc. Int. Conf. Body Area Netw.*, pp. 1–8, 2008.

[23] V. Shnayder, B. Chen, K. Lorincz, T. Fulford-Jones, and M. Welsh, "Sensor networks for medical care," Harvard Univ., Cambridge, MA, Tech. Rep. TR-08-05, 2005.

[24] J. P. M. G. Linnartz, J. A. Groot, J. J. Lukkien, and H. Benz, "A novel architectural concept for trustworthy and secure access to body sensor information," in *Proc. Int. Conf. Intell. Sys. Knowledge Eng.*, pp. 417–423, 2009.

[25] O. G. Morchon, H. Baldus, and D. S. Sanchez, "Resource-efficient security for medical body sensor networks," in *Proc. Body Sensor Netw.*, 2006, pp. 80–83.

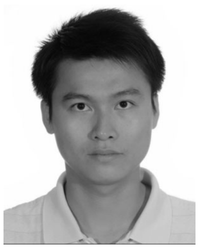
[26] A. Hande, T. Polk, W. Walker, and D. Bhatia, "Self-powered wireless sensor networks for remote patient monitoring in hospitals," *Sensors*, vol. 6, no. 9, pp. 1102–1117, 2006.

[27] Q. Zhang, T. Yu, and P. Ning, "A framework for identifying compromised nodes in wireless sensor networks," *ACM Trans. Inf. Syst. Security*, vol. 11, no. 3, pp. 1–37, 2008.

[28] C. C. Tan, H. Wang, S. Zhong, and Q. Li, "IBE-lite: A lightweight identity-based cryptography for body sensor networks," *IEEE Trans. Info. Tech. Biomed.*, vol. 13, no. 6, pp. 926–932, 2009.

[29] D. He, C. Chen, S. Chan, and J. Bu, "SDRP: A secure and distributed reprogramming protocol for wireless sensor networks," *IEEE Trans. Ind. Electron.*, 2012, DOI: 10.1109/TIE.2011.2178214.

[30] TinyOS: An open-source OS for the networked sensor regime [Online]. Available: http://www.tinyos.net/



Daojing He received the B.Eng. and M.Eng. degrees in computer science from Harbin Institute of Technology, Heilongjiang, China, in 2007 and 2009, respectively. He is currently working toward the Ph.D. degree in the Department of Computer Science, Zhejiang University, Hangzhou, China.

His research interests include network and systems security with focuses on wireless security.

He is the Technical Program Committee member of the IEEE Globecom 2011, the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications 2011, the IEEE International Conference on· Communications 2012, the IEEE Wireless Communications and Networking Conference 2012, etc.

Chun Chen received the Bachelor's degree in mathematics from Xiamen University, Fujian, China, in 1981, and the Master's and Ph.D. degrees in computer science from Zhejiang University, Hangzhou, China, in 1984 and 1990, respectively.

He is currently a Professor at the College of Computer Science, and the Director of the Institute of Computer Software at Zhejiang University. His research interests include image processing, computer vision, and embedded system.

Sammy Chan received the B.E. and M.Eng.Sc. degrees in electrical engineering from the University of Melbourne, Melbourne, Vic., Australia, in 1988 and 1990, respectively, and the Ph.D. degree in communication engineering from the Royal Melbourne Institute of Technology, Melbourne in 1995.

From 1989 to 1994, he was with Telecom Australia Research Laboratories, first as a Research Engineer, and between 1992 and 1994 as a Senior Research Engineer and Project Leader. Since December 1994, he has been with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong, SAR, where he is currently an Associate Professor.

Jiajun Bu received the B.S. and Ph.D. degrees in computer science from Zhejiang University, Hangzhou, China, in 1995 and 2000, respectively.

He is currently a Professor in the College of Computer Science and the Deputy Dean of the Department of Digital Media and Network Technology, Zhejiang University. His research interests include embedded system, mobile multimedia, and data mining.

Athanasios V. Vasilakos received the B.S. degree in electrical and computer engineering from the University of Thrace, Xanthi, Greece, in 1983; the M.S. degree in computer engineering from the University of Massachusetts, Amherst, in 1986; and the Ph.D. degree in computer engineering from the University of Patras, Patras, Greece, in 1988.

He is currently a Professor at the University of Western Macedonia, Florina, Greece. He has authored or co-authored more than 200 technical papers in major international journals and conferences. He is author/coauthor of five books and 20 book chapters in the areas of communications.

Prof. Vasilakos was the General Chair and Technical Program Committee Chair of many international conferences. He is an Editor and Guest Editor for many technical journals. He is founding Editor-in-Chief of the *International Journal of Adaptive and Autonomous Communications Systems* and the *International Journal of Arts and Technology*. He is also the General Chair of the Council of Computing of the European Alliances for Innovation.