A Distributed Trust Evaluation Model and Its Application Scenarios for Medical Sensor Networks

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

***Abstract*—The development of medical sensor networks (MSNs) is imperative for e-healthcare, but security remains a formidable challenge yet to be resolved. Traditional cryptographic mechanisms do not suffice given the unique characteristics of MSNs, and the fact that MSNs are susceptible to a variety of node misbehaviors. In such situations, the security and performance of MSNs depend on the cooperative and trust nature of the distributed nodes, and it is important for each node to evaluate the trustworthiness of other nodes. In this paper, we identify the unique features of MSNs and introduce relevant node behaviors, such as transmission rate and leaving time, into trust evaluation to detect malicious nodes. We then propose an application-independent and distributed trust evaluation model for MSNs. The trust management is carried out through the use of simple cryptographic techniques. Simulation results demonstrate that the proposed model can be used to effectively identify malicious behaviors and thereby exclude malicious nodes. This paper also reports the experimental results of the Collection Tree Protocol with the addition of our proposed model in a network of TelosB motes, which show that the network performance can be significantly improved in practice. Further, some suggestions are given on how to employ such a trust evaluation model in some application scenarios.**

***Index Terms*—Medical sensor networks (MSNs), network performance, privacy, security, trust evaluation.**

I. Introduction

RECENTLY, with the rapid development in wearable medical sensors and wireless communication, wireless medical sensor networks (MSNs) have emerged as a promising technique that will revolutionize the way of seeking healthcare [1], which is often termed as e-healthcare. Instead of being measured face-to-face, a patient's health status can be sensed remotely, continuously, and in real time, and then processed and transferred to a hospital or healthcare center for monitoring.

Obviously, the critical nature of MSNs stems from their applications in life-saving infrastructures such as medical monitoring of victims in emergency scenarios and long-term monitoring of diseased patients. The lack of adequate security features may not only lead to a breach of patient privacy, but also potentially allow attackers to modify actual data, resulting in wrong diagnosis and treatment. Over the years, some mechanisms have been proposed for securing MSNs (e.g., [2]–[5]). The security challenges facing a wireless sensor network (WSN) for wireless health monitoring have been identified, and a security architecture called "SNAP" has been proposed in [2]. A lightweight security system has been proposed in [3], which allows distributed key establishment and access control in MSNs. Very recently, a lightweight identity-based cryptography named IBE-Lite has been presented in [4], where identity-based public key is used to encrypt all medical data. It balances security and privacy with accessibility. Later, a novel and privacy-preserving distributed access control scheme for sensor networks is presented in [5].

However, all these protocols are only based on cryptographic techniques to achieve security. It should be noted that traditional cryptographic mechanisms do not suffice given the unique characteristics of MSNs, and the fact that MSNs are susceptible to a variety of node misbehaviors. A compromised/malicious node may launch an attack on patient privacy. A faulty node might have a software fault that prevents it from behaving normally. Obviously, misbehaving nodes cannot be tackled by cryptographic techniques alone. Therefore, even if all these approaches (e.g., [2]–[5]) are employed, medical records may be modified freely by the attackers, and false information can be injected by a compromised node. Also, the resource constraints of medical sensor nodes make the use of many solutions including asymmetric cryptosystems unrealistic or impossible in the majority of circumstances. Further, these approaches cannot pinpoint exactly from where the false information is introduced into the network and who is responsible for it. In the literature, one viable solution to solve the aforesaid issues is to use trustworthiness to identify malicious/faulty nodes and thereby exclude them from an MSN. However, as to be discussed in Section II, little work focuses on application-independent trust evaluation for wireless networks, not to mention specifically for an MSN. As a result, there is a growing demand for application-independent and distributed trust evaluation models for MSNs.

This paper makes three main contributions.

1) We develop an application-independent and distributed trust evaluation model in MSNs, which allows each node

Manuscript received November 9, 2011; revised March 4, 2012; accepted May 5, 2012. Date of publication May 17, 2012; date of current version November 16, 2012. This work was supported in part by the Scholarship Award for Excellent Doctoral Student granted by the Ministry of Education, National Science Foundation of China (Grant 61070155), in part by the Program for New Century Excellent Talents in University (NCET-09-0685), and in part by the Research Grants Council of the Hong Kong, SAR (Project No. 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, Zhejiang 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.2012.2199996

to perform trust evaluation based on continuous monitoring of the behaviors of its neighbors, in order to improve security and performance of MSNs. To the best of our knowledge, this is the first application-independent trust evaluation model for MSNs. The proposed model has the following advantages. First, different from most existing trust evaluation works, we argue that trust evaluation model and simple cryptographic techniques are needed to complement each other as neither of them alone can provide a solution to ensure the security and privacy of MSNs. Second, we identify the unique features and security requirements of MSNs and then introduce relevant node behaviors, such as transmission rate and leaving time, into trust evaluation to detect malicious, compromised, and faulty nodes for the first time in the literature. Also, our proposed model employs different mathematical functions for different behaviors based on their characteristics. Third, our approach is to allow the nodes to develop a community of trust. In the proposed model, each node manages trust records of other nodes about performing some activities, which are used as an inherent aspect in predicting their future behaviors. Not only can such a mechanism identify malicious behaviors and thereby exclude malicious nodes, but also improve network performance because honest nodes can avoid working with less trustworthy nodes. Fourth, it is very simple for each node to implement such a trust evaluation system. Therefore, the proposed model is particularly suitable for MSNs comprising resource-constrained medical sensor nodes.

2) By simulations, we demonstrate that the proposed model can be used to effectively identify malicious behaviors and exclude malicious nodes. We also implement the Collection Tree Protocol (CTP) [6] with the addition of our proposed model in a testbed of resource-limited motes. Evaluation results show that compared to existing trust evaluation approaches, the proposed model can significantly improve the average packet reception ratio (PRR) in practice.
3) Further, some application scenarios, such as centralized malicious node detection and secure unicast routing, are provided to illustrate how to employ such a trust evaluation model.

The remainder of this paper is organized as follows. Section II gives an overview of related works and features of MSNs. The network and threat models are summarized in Section III. Our proposed trust evaluation model is described in Section IV. Section V provides security analysis, performance, and functionality evaluations of the proposed model. Then in Section VI, two examples of application scenarios are given to demonstrate how to employ such a trust evaluation model. Section VII concludes this paper.

II. Related Works and Features of MSNs

A. Related Works

The research on trust evaluation has been extensively performed for a wide range of applications in the Internet, including e-commerce [7] and peer-to-peer network [8]. However, empirical studies of wireless security have demonstrated that traditional strategies for network security that are applicable to wired networks do not work well in wireless networks due to the special characteristics of wireless communications [9]. On the other hand, although establishing trust among distributed network entities has been recognized as a powerful tool to secure MANETs (e.g., [10]–[14]) and WSNs (e.g., [15]), little work focuses on application-independent trust evaluation for wireless networks. For example, the authors of [10] and [11] focus on secure routing, and the authors of [15] just consider secure location discovery. Moreover, in most trust approaches (e.g., [13], [14]), only linear functions have been used to compute a node's trust value.

To our knowledge, *TrE* [9] is the only trust evaluation model related to an MSN, which is proposed for secure multicast. By simulations, the authors have shown that the security and efficiency of *TrE* are better than currently accepted trust schemes (e.g., [12]). However, in *TrE*, only the residential time and historical trust records of a node are considered. Moreover, each node only relies on its direct monitoring for calculating trust value, which makes it vulnerable against collaborative attacks. Also, *TrE* only considers successful packet forwarding but does not take failed packet forwarding into account. Therefore, *TrE* is too simple to ensure the security and efficiency of MSNs. Most importantly, we observe that since all these works ([7]–[15]) do not consider the unique operational and security requirements of MSNs, they might not be suitable for MSNs.

B. Unique Features of MSNs

In this section, some differences between MSNs and MANETs (or WSNs) are listed as follows [1].

1) *Data rate*: Many MANETs and WSNs are employed to monitor events which often happen at irregular interval. On the other hand, MSNs are employed for monitoring human's physiological activities and actions, which may occur in a more periodic manner. Hence, the applications' data rates are relatively more steady.
2) *Mobility:* Nodes in an MSN are either static (e.g., the nodes in the hallways) or relatively static (e.g., all sensor nodes in the same person).
3) *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 nodes of MSNs is much easier than those in WSNs, which can be physically unreachable after deployment. Thus, it may not be necessary to maximize battery lifetime in an MSN at the expense of higher latency.

III. Network and Threat Models

A. Network Model

As shown in Fig. 1, an MSN is a health monitoring network of sensor nodes attached to patients and healthcare sites. We assume that the sensor nodes communicate through the

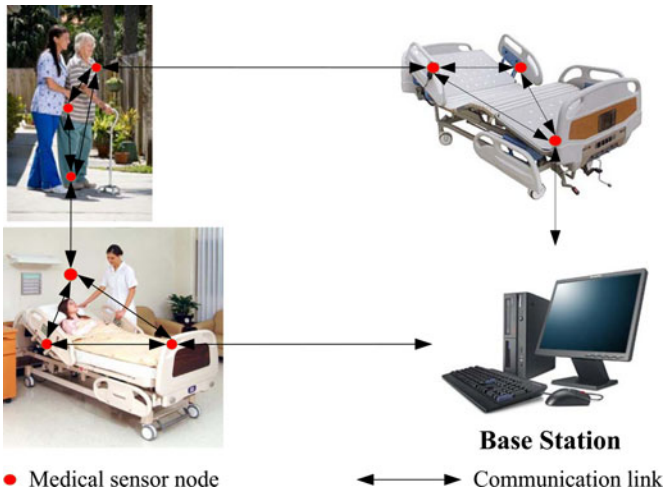


Fig. 1. Schematic diagram of an MSN.

wireless medium, as wires running between sensors in an MSN will make it obtrusive, especially in the case of implanted and placed sensors or when there is a need to reconfigure the placement of sensors on the body. In this paper, we use the terms nodes, sensors, and sensor nodes interchangeably. Two sensors can communicate directly with each other when they are within their transmission ranges. Additionally, other sensors can cooperate by functioning as routers to relay the exchanged information. Thus, these wearable devices, placed nodes, and medical monitoring sensors form a multihop wireless network which does not need any predeployed infrastructure. We assume that there is a loose time synchronization among the sensor nodes with the help of some existing secure time synchronization schemes (e.g., flooding time synchronization protocol (FTSP) [16]).

B. Threat Model

Due to the sensitive nature of the data MSNs collect and the broadcast nature of the wireless communication environment, MSNs potentially face many threats. They are imposed by either active or passive attackers. Active attackers can modify messages, inject forged messages, replay old messages, compromise nodes, and spoof nodes in the MSN in order to become part of the network. Also, in active attack, a compromised/malicious node would drop messages and launch denial-of-service (DoS) attacks by sending a large volume of bogus packets to jam the communication channels. Active attackers not only invade patient 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. For passive attackers, they do not interfere with the functions of the MSN, but can intercept exchanged messages within an MSN by eavesdropping. By offline cryptanalysis of intercepted confidential data, passive attackers can invade patient privacy.

IV. Proposed Trust Evaluation Model

A. Overview of Trust Evaluation Methodology for MSNs

A *trust relationship* is always established between two nodes for a specific action. That is, one node trusts the other node about performing some actions. In this study, the first node is referred to as the *subject* and the second node as the *agent*. A notation {subject: agent, action} is introduced to represent a trust relationship. For each trust relationship {subject: agent, action}, a numerical value T(subject: agent, action), referred to as *trust value*, describes the level of trustworthiness.

Trust is evaluated both independently by each node based on its observations, and cooperatively through sharing recommendations and spreading reputation. More specifically, in MSNs, trust relationship can be established in three ways. First, for each time unit, when the subject can directly observe the agent's behavior, *direct trust* can be established. Second, when the subject cannot directly observe the agent's behavior, it can receive recommendations from qualified nodes about the agent, which are their trust values for the agent. Thus, *recommended trust* can be established. Third, when the subject uses the trust records of the agent for the previous time units, *historical trust* can be established. In the following, we assume that node *A* is the *subject* and node *B* is the *agent*.

In order to establish the trust relationship mentioned previously, we use the concept of *community of a node* [9] which includes the node itself, referred to as the central node, and all of its one-hop neighbors, among which some may be malicious. Often, as a multihop network, an MSN employs some protocols which periodically broadcast short HELLO messages (called beacon messages) within its one-hop area. For example, HELLO messages are widely used for neighbor discovery in routing protocols (e.g., Ad hoc On-Demand Distance Vector (AODV) routing protocol [17], and optimized link state routing protocol) for wireless multihop networks. Also, because of the difficulty of obtaining IEEE 802.11 feedback about link connectivity in real networks, many current protocol implementations such as FTSP [16] utilize HELLO messages. These HELLO messages can be used to maintain each node's community. More exactly, a node keeps track of its one-hop neighbors simply by listening to their HELLO messages.

A simple and efficient trust evaluation methodology is given as follows.

1) As the central node of its community, each node observes some behaviors of its one-hop neighbors during each time unit and then generates the direct trust of each of these nodes about performing a specific behavior for the time unit. The size of each time unit is predefined according to the specific application, which is a multiple of the time interval between two HELLO messages.
2) Obviously, each node cannot directly observe the behaviors of its non-one-hop neighbors. Suppose that a node, say node A, wants to establish trust relationships with its non-one-hop neighboring nodes within its q hops area ($q > 1$) about performing a specific behavior for each time unit, the set of which is denoted by Ψ. Node A first checks the trust values of all of its one-hop neighbors and selects

a set of nodes which have the trust values larger than a threshold (i.e., the most highly trusted nodes). The threshold value is predefined according to the specific application. The qualified nodes can deliver the recommended trusts of Ψ to node *A*. Thus, each node can get the recommended trust values of its non-one-hop neighboring nodes about performing a specific behavior for the time unit. Note that compared to the time at which the direct trust value is calculated, the calculation of the recommended trust values should be carried out after a short period. More specifically, the recommended trust values should be calculated only after the total trust values of the central node's one-hop neighbors have become stable.

3) A node's trust values for the previous time units (i.e., historical trust values) are taken into account in order to measure its trustworthiness for the current time unit. That is, with the direct (respectively, recommended) trust value and the historical trust value of each one-hop neighboring node (respectively, non-one-hop neighboring node) as input, each node can calculate the integrated trust values of other nodes associated with a specific behavior for each time unit.
4) Finally, with these integrated trust values, each node calculates the total trust values of other nodes for each time unit.

The susceptibility to node misbehaviors can affect the trust evaluation model itself. Especially for models that require cooperative trust evaluation, it is crucial that the nodes are willing to cooperate by providing recommendations or evidences that they may hold for the target node. This is the case in MSNs, since all nodes belong to the same authority (i.e., the network owner). This can be achieved by network configuration during the setup and operation phases of an MSN. However, this is not the case in MANETs, since different nodes may belong to different authorities (i.e., different network users), which may behave selfishly to preserve resources. Thus, compared to MANETs, trust evaluation is more applicable for MSNs.

B. Employed Cryptographic Technique

The proposed trust management is carried out through the use of simple cryptographic techniques such as symmetric encryption/decryption and hash operations. Our system requires three types of keys for each node—an individual key shared with the base station, a pair-wise key shared with each of its one-hop neighbors, and a multicast key shared with a group of nodes. Also, there is a global key that is shared by all the nodes in the network. The protocol used for establishing and updating these keys is both communication- and energy-efficient, and minimizes the involvement of the base station. This can be achieved by some simple and efficient methods (e.g., predeployment-based approaches [18] and communication-based approaches [19]). There are typically three types of communication patterns in an MSN: unicast (addressing a message to a single node), multicast (addressing a message to multiple neighboring nodes), and global broadcast (addressing a message from the base station to all the nodes in the network). The detailed description is given as follows.

1) Through the individual key, each node can encrypt its sensed readings and transmit the encrypted message to the base station via multihop routing. Thus, an MSN does not leak sensor readings to anyone except the base station. Also, a node may send an alert to the base station over a multihop path if it observes that the trust value of a neighboring node is less than a predefined threshold. Similarly, the base station can use this key to encrypt any sensitive information, e.g., keying material for key updating, that it needs to send to an individual node.
2) For each node, a pair-wise key shared with one of its one-hop neighbors can be used for securing its message. Only the neighbor uses the same key for decrypting or verifying its message.
3) The multicast key enables a node to multicast messages to a group of nodes in a secure manner.
4) With the global key, the base station can encrypt a message and then broadcast it to the whole network. For example, the base station issues missions, sends queries, and interests. Since the global key is shared among all the nodes in the network, an efficient rekeying mechanism is necessary for updating this key after a compromised node is revoked.

Here, we denote $h(.)$ to be a public one-way hash function and $E_K(.)$ to be the symmetric encryption with the secret key K. We consider that node *A* wants to deliver a message *msg* to a group of nodes Υ (respectively, the base station). In this case, node *A* needs to encrypt *msg* as $E_{K_{AG}}(msg \| h(msg))$ [respectively, $E_{K_{AS}}(msg \| h(msg))$], where K_{AG} (respectively, K_{AS}) is the multicast key shared with nodes Υ (respectively, an individual key shared with the base station). Note that a timestamp or nonce is added into *msg* to resist the replay attacks. Obviously, the key can ensure the confidentiality and authenticity of *msg* while $h(msg)$ can be used to ensure the integrity of *msg*.

C. Obtaining Direct Trust

Direct trust is established upon observations on whether the previous behaviors of the agent are honest. Note that any behavior in which the underlying detection system is interested can be chosen. According to some features of MSNs such as data rate and mobility, some relevant node behaviors such as leaving time and transmission rate are introduced into the design of the proposed model.

Motivated by the fact that the observations of packet forwarding can be used to identify the compromised or malicious nodes, packet forwarding is used in the trust evaluation. Consider in each time unit, as the central node of its community, each node, say *A*, asks one of its one-hop neighbors, say node *B*, to forward N packets, and B in fact forwards k packets, where $k \leq N$. According to the beta-function-based method [20], the direct trust value $T(\text{A: B, packet forwarding})$ is calculated as $\frac{k+1}{N+2}$. When k is close to N, the direct trust value should decrease slowly with the decrease of k. That is because node *A* cannot very exactly observe the number of packets forwarded by node *B* in a wireless channel. However, when k is very small compared

to N, which means node B drops a large number of packets and is very likely to be malicious, the direct trust value should decrease fast with the decrease of k. Of course, due to poor channel conditions, node B may drop a large number of packets. In this case, direct trust value should decrease fast; thus, the administrator can find such an abnormal node. Compared with exponential and linear functions, logarithmic functions have a fast increase shape when the parameter is not a large number, and a slow increase shape when the parameter is a large number. Therefore, here logarithmic functions are employed. $T(\text{A: B, packet forwarding})(=T^f_{AB})$ is calculated as follows:

$$T^f_{AB} = \log_2\left(1 + \frac{k+1}{N+2}\right). \tag{1}$$

Obviously, $T^f_{AB} \approx 1$ when $k = N$ while $T^f_{AB} \approx 0$ when $k \ll N$.

As described in Section II-B, different from most MANETs or WSNs, most nodes remain static or relatively static. More exactly, the set of one-hop neighbors of each node often remain unchanged. As the central node of its community, each node, say A, can estimate the leaving time of its one-hop neighboring node B as $t\_l = T2 - T1$, where $T1$ is node A's system time when receiving the latest message from node B, and $T2$ is node A's current time. As described in Section IV-A, in order to build and maintain a node's community, each node needs to periodically broadcast HELLO messages within its one-hop area. Obviously, through this procedure, the observation of neighbors' leaving time can also be achieved. It should be noted that a node compromise attack often consists of three stages [21]. The first stage is physically obtaining and compromising the sensors; the second stage is redeploying the compromised nodes back to the network; and in the last stage, the compromised nodes rejoin the network and launch attacks. Therefore, through the use of leaving time, compromised nodes may be identified before the redeployment stage and prevented from rejoining the network. Note that a node of MSNs may be temporarily disabled (e.g., channel problems or temporary unavailability) and should not be judged as a malicious node. Thus, when t_l is small, the trust value should fall slowly with the increase of t_l. With the increase of t_l, node B is more and more likely to be a malicious (or abnormal) node; thus, the trust value should decrease faster. Compared with logarithmic functions and linear functions, exponential functions have a small gradient when the input value is small, and a large gradient for a large input value. Thus, here exponential functions are employed. In every community, the central node should set a *Leaving Time Threshold* δ according to the specific application. For example, δ can be set to several intervals of HELLO messages. Once the leaving time of node B is more than δ, the direct trust is set to 0. As shown later, node A calculates the direct trust value of node B associated with the leaving time as $T(\text{A: B, leaving time})(=T^l_{AB})$, where $a > 1$ and $a^\delta = 2$:

$$T^l_{AB} = \begin{cases} 2 - a^{t\_l} & \text{if } t\_l < \delta \\ 0 & \text{else.} \end{cases} \tag{2}$$

As shown in Fig. 2, the gradient of the leaving-time trust value depends on the value of δ. That is, the larger the value of δ, the slower the trust about leaving time decreases. Thus, if a node is located in a stationary environment with good channel condition (e.g., in ceiling), δ should be set to a small value; otherwise, δ should be set to a big value.

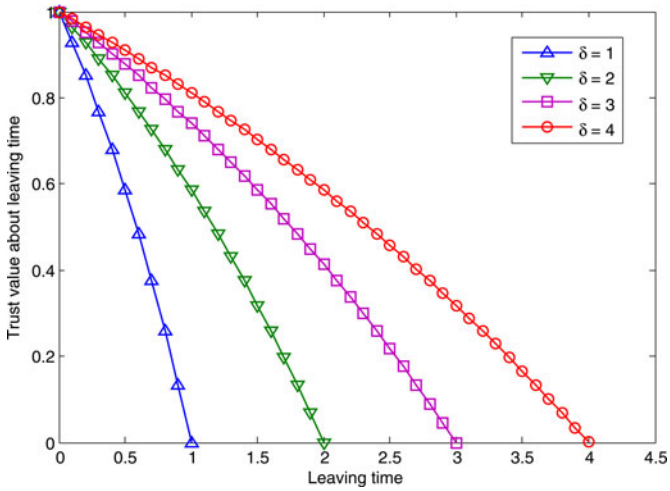


Fig. 2. Trust value about leaving time.

As described in Section II-B, different from most MANETs or WSNs, the data streams in an MSN exhibit relatively stable rates. More exactly, the transmission rate of a specific node is limited to the range $[\theta_1, \theta_2]$. Because the data transmission rate of various biosensors is heterogeneous (e.g., the data rate of a blood pressure sensor is about 1.2 kb/s while that of a body temperature sensor is about 0.0024 kb/s [22]), the thresholds θ_1 and θ_2 are predefined according to the targeted sensor, i.e., $\theta_1 < 1.2\text{kb/s} < \theta_2$ for the blood pressure sensor. Also, if the transmission rate of a specific sensor is stable, the range $[\theta_1, \theta_2]$ should be small; otherwise, the range should be wider. Motivated by the fact that the observation of transmission rate can be used to detect the compromised or malicious nodes, the transmission rate is introduced into the trust evaluation. Assume that in each time unit, node A observes that the current transmission rate of node B is t_r. As illustrated in Fig. 3, when t_r is a little smaller than θ_1, node B's trust value should decrease slowly with the decrease of t_r. That is because node A cannot very exactly observe the transmission rate of node B in a wireless channel. However, if t_r is much lower than θ_1, node B's trust value should decrease fast with the decrease of t_r. When t_r is larger than θ_2, which means that node B must have transmitted extra bogus packets, the decrease speed of its trust value should immediately become fast. According to the aforementioned analysis, T^t_{AB} which indicates the trust value of $\{\text{A: B, transmission rate}\}$ is calculated as follows, where $b > 1$ and $0 < c < 1$. Here, b and c are set to 20 and 0.11, respectively.

$$T^t_{AB} = \begin{cases} \log_b\left(1 + (b-1)\dfrac{t\_r}{\theta_1}\right) & \text{if } t\_r < \theta_1 \\ 1 & \text{else if } \theta_1 \le t\_r \le \theta_2 \\ c^{\frac{t\_r - \theta_2}{\theta_2}} & \text{else.} \end{cases} \tag{3}$$

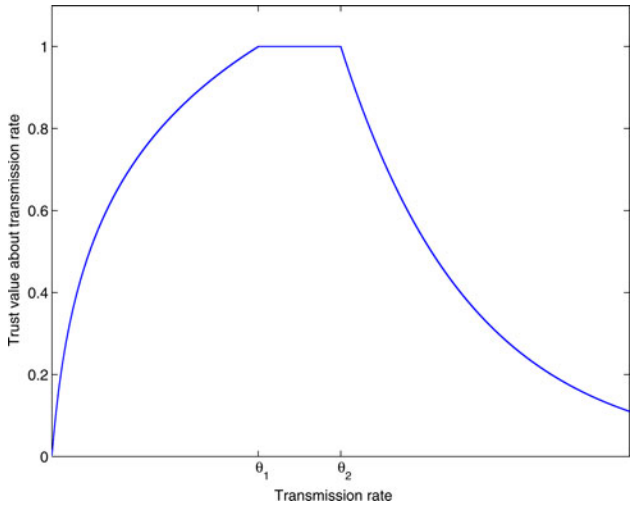


Fig. 3. Trust value about transmission rate.

D. Obtaining Recommended Trust

Requiring recommended trust in MSNs often occurs in the circumstance where the subject cannot directly observe an agent's behaviors. That is, the agent is not a one-hop neighboring node of the subject. Obviously, in this case, direct trust cannot be established. Assume that in each time unit, node *A* wants to establish trust relationships with its non-one-hop neighbors within its q-hop area ($q > 1$), denoted by $\Psi = \{Q_1, Q_2, \ldots\}$, about action *act*. The value of q should be set according to the resources (e.g., storage space and power) of each node in the MSN. Here, we consider $q = 2$ as an example. Note that the same procedure can be used for the case where $q > 2$. The recommended trust value $T^r_{AB_i}(=T(A : B_i, act))$ can be obtained as follows.

Node *A* first checks the trust values of all of its one-hop neighbors and selects a set of nodes, denoted by Ω, which have the trust values [these trust values can be calculated according to (9)] larger than a threshold. Subsequently, node *A* encrypts the requests for recommended trusts of Ψ using the multicast key and then takes advantage of the multicast mechanism to send the encrypted information to each of Ω. Here, we assume that $\forall B \in \Omega, \forall C \in \Psi$ and *C* is a one-hop neighboring node of node B. Node *B* observes the behavior of node *C* and makes recommendation to node *A* as $T_{BC} = T(B : C, \text{act})$. Note that node *B* delivers T_{BC} after encrypting it using its pair-wise key with node *A*. We assume that node *A* trusts node *B* about performing the recommendation with R_{AB}. Here, R_{AB} is set to the total trust value of node *B* in node *A*'s point of view, which can be calculated by (9). T_{BC} is set to the integrated trust value of node C in node B's point of view about performing a specific behavior, which can be calculated by (7). We assume that node *A* calculates the recommended trust value of node *C* associated with behavior *act* as $T_{AC} = T(A : C, act)$. Thus,

$$T_{AC} = R_{AB} \times T_{BC}. \tag{4}$$

Obviously, a given node (say *A*) discounts the trust value for a node (say *C*) recommended by its one-hop trusted neighbors. Similarly, node *A* trusts node B_i about performing the recommendation with R_{AB_i}; let $T_{B_iC} = T(B_i : C, \text{act})$, where $i \in \{1, \ldots, n\}$, $B_i \in \Omega, \forall C \in \Psi$ and *C* is a one-hop neighboring node of node B_i. Therefore, node *A* can establish trust to node *C* through n paths: *A*–B_i–*C*. To combine the trust established through different paths, we employ the maximal ratio combining as

$$T(A : C, act) = \sum_{i=1}^{n} \omega_i (R_{AB_i} \times T_{B_iC}) \tag{5}$$

where $\omega_i = \frac{R_{AB_i}}{\sum_{i=1}^{n} R_{AB_i}}$.

E. Obtaining Historical Trust

Note that node *A* has made observations and then calculates the direct (respectively, recommended) trust values of each one-hop neighbor (respectively, non-one-hop neighbor) for the previous time units. This procedure has been described in Sections IV-C and IV-D. We assume that at previous time unit t_j, node *A* observes and then calculates the trust value of node *B* about performing action *act* as $T_j(A : B, act)$, where $j = 1, 2, \ldots, m$. Here, m should be chosen according to the specific application scenario. We propose to calculate the historical trust value $T^h_{AB}(=T^h(A : B, act))$ as follows:

$$T^h_{AB} = \frac{\sum_{j=1}^{m} \beta_j \times T_j(A : B, act)}{\sum_{j=1}^{m} \beta_j}. \tag{6}$$

Here, we introduce $0 \le \beta_j \le 1$ as the aging factor, which describes that the trust value made long time ago should be less important than the trust value made more recently. Therefore, $\beta_1 < \beta_2 \ldots < \beta_m$. When node *B*'s behavior changes fast, the observations made long time ago is not very useful for predicting node *B*'s future behavior. In this case, β_j should be a small value, and vice versa. The use of the aging factor provides a way to capture dynamic changes in node *B*'s behaviors.

F. Obtaining Integrated Trust Value About Performing an Action

As described in Sections IV-C, IV-D, and IV-E, for each time unit, node *A* has established the direct (respectively, recommended) trust value $T^y(A : B, act)$ and the historical trust value $T^h(A : B, act)$ (for the previous time units) of each one-hop neighboring node (respectively, non-one-hop neighboring node) *B* about behavior *act*. To take into account these trust values, in current time unit, node *A* calculates the integrated trust value of node *B* about performing *act* as follows:

$$T(A : B, act) = \alpha \times T^y(A : B, act) + \gamma \times T^h(A : B, act) \tag{7}$$

where $0 < \alpha < 1$, $0 < \gamma < 1$ and $\alpha + \gamma = 1$.

G. Calculating the Total Trust

As described previously, for each time unit, each node, say *A*, has established the trust values of other nodes (say *B*) about performing multiactions. Here, we assume that node *A* has established the integrated trust value $T(A : B, act_j)$ of node *B* about performing action act_j, where $j \in \{1, 2, \ldots, p\}$. With

$T(A:B, act_j)$ as input, node *A* calculates the total trust value T_{AB}^{total} of node *B* associated with these multiactions as follows:

$$T_{AB}^{\text{total}} = f(T(A:B, act_1), \ldots, T(A:B, act_p)). \quad (8)$$

Here, $f(.)$ is an application-dependent function. 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. Therefore, different weights are introduced into (8). Thus

$$T_{AB}^{\text{total}} = \epsilon_1 \times T(A:B, act_1) + \cdots + \epsilon_p \times T(A{:}B, act_p) \quad (9)$$

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. For example, since the aim of secure unicast routing is that the messages from the source node can be successfully forwarded to the destination node, we should give a larger weight to the packet-forwarding trust value T(A: B, packet forwarding) when the proposed system is applied in secure unicast routing. These weights can be loaded on each sensor node during the system initiation phase.

When the proposed trust model is implemented on each sensor node, the trust value calculated by the aforementioned equations should be mapped to an integer in $[0, 100]$, where 0 denotes the most untrusted state while 100 denotes the most trusted state. This representation of trust gives an MSN many benefits in terms of computation and memory usage, transmission, and reception power. Also, the logarithmic function (or exponential function) could be evaluated as polynomials through Taylor's expansion. The maximum order of Taylor's expansion is set to balance the accuracy of the function and the resource of nodes of the MSN.

V. Security Analysis, Performance, and Functionality Evaluations

In this section, we first analyze the security and communication overhead of the proposed trust model. Then we carry out some simulations and real experiments to evaluate the functionality of our system.

A. Security Analysis

Here, it is demonstrated that, because trust model and simple cryptographic technique complement each other, our proposed model can efficiently ensure the security of MSNs. Throughout this paper, the proposed simple cryptographic techniques have been incorporated into the design of our trust model to ensure the confidentiality, integrity, and authenticity of transmitted data in an MSN. Thus, the proposed model can prevent the attackers from eavesdropping, modifying, deleting, injecting, and replaying messages.

However, traditional cryptographic techniques (e.g., public key and symmetric cryptography) are susceptible to a variety of node misbehaviors (e.g., some nodes have been compromised or a malicious/faulty node launches DoS attacks). Note that our trust model solves these security and privacy issues which traditional cryptographic techniques cannot deal with. Each node can manage the trust records of other nodes and detect faulty, compromised, or malicious nodes. The prediction of nodes' future behavior directly determines the risk faced by the MSN. Given the risk, the MSN can adapt its operation accordingly. For example, stronger security mechanisms should be employed when risk is high. Also, with the assessment of trustworthiness of individual network entities, it is possible to evaluate the trustworthiness of the MSN. For example, the distribution of the trust values of network entities can be used to represent the healthiness of the MSN. Further, the proposed trust model can pinpoint exactly which node of an MSN is malicious and thereby exclude it. The detailed analysis is given as follows.

The trust management about transmission rate can not only defeat DoS attacks but also prevent the attackers from dropping messages. Also, through the trust evaluation associated with leaving time, compromised nodes may be identified before the redeployment stage and prevented from rejoining the network. Additionally, secure routing using the proposed trust model can monitor route disruption in MSNs and adjust route selection dynamically and thus prevent malicious nodes from dropping packets. Further details will be given in Section V-C2.

B. Performance Evaluation

The communication overhead is measured as the total number of packets transmitted by all the nodes for trust management, which is also related to power consumption. As described in Section IV, the observations of packet forwarding and transmission rate do not incur any communication overhead. The observations of leaving time are measured by each node through listening to the HELLO messages from one-hop neighboring nodes. Since, in general, an MSN employs some protocols which periodically broadcast HELLO messages, such observations do not introduce any communication overhead. As a result, communication overhead is only incurred in the second step, i.e., obtaining recommended trust. To analyze communication overhead of this step, we consider $q = 2$ as an example. As described in Section IV-D, each node, say *A*, selects some qualified nodes Ω and then multicasts the request message for recommended trusts to Ω, where the request message indicates the identities of node *A*'s two-hop neighboring nodes. Upon receiving this message, those qualified nodes will give a response to node *A*. We assume that the number of nodes in an MSN is M, $|\Omega| = r$ and the rate of sending request message is μ_r packets per second. Therefore, the communication overhead of the proposed trust model is $\mu_r \times M \times (1 + r)$ packets per second. Note that these parameters should be set to achieve a tradeoff between security and performance, i.e., good security strength while requiring moderate communication overhead. Compared to our proposed model, *TrE* also requires the use of HELLO messages, but has no recommendation trust mechanism; its communication overhead is smaller. However, due to its simple way of establishing trust, *TrE* provides relatively weaker security protection.

To verify the aforesaid analysis, we have implemented a real-world experiment with multiple TelosB motes and different

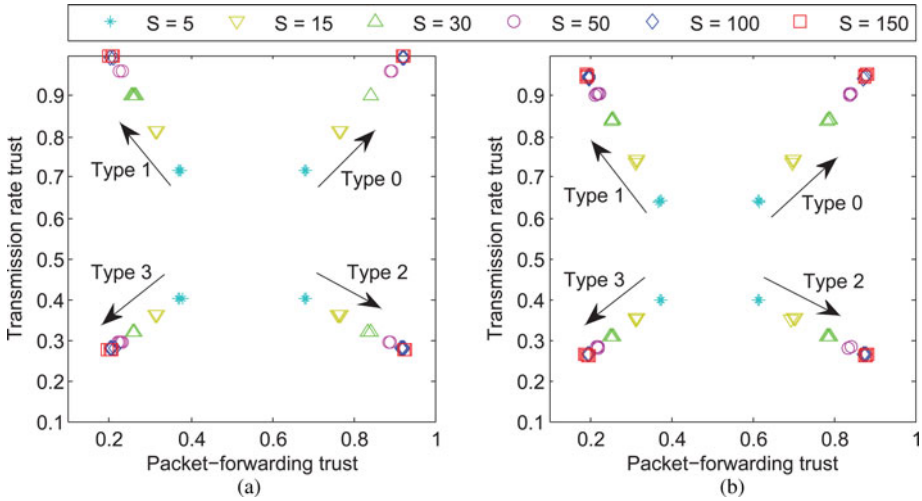


Fig. 4. Trust record in a legitimate node about its neighbors' packet forwarding and transmission rates. (a) One-hop neigboring nodes. (b) Two-hop neigboring nodes.

values of $|\Omega|$. The TelosB mote has an 8 MHz CPU, 10 kb RAM, 48 kb ROM, and an 802.15.4/ZigBee radio. The motes run TinyOS 2.1.0. In our implementation, the distance between a center node and its neighbors is 1 m, and the radio power level of each node is set with default value. Here, we assume a packet size of 7 bytes, and each packet consists of source ID (2 bytes), destination ID (2 bytes), group ID (1 byte), trust value (1 byte), and CRC (1 byte). The rate of sending request packet is 0.5 packets per second. The rate of sending the response packet is based on the random sending mechanism which is about 0.67 packets per second. Each experiment lasts for 500 s. The communication overheads predicted by the aforementioned analysis are 2.647, 5.285, 10.509, and 19.522 packets per second when $|\Omega|$ equals to 2, 4, 8, or 15, respectively. The corresponding results of our experiment are 2.647, 5.288, 10.529, and 19.623 packets per second, respectively. Thus, the experimental results match the predicted results.

C. Functionality Evaluation

The functionalities of the proposed trust model are evaluated either by simulations or real experiments, and the results are presented and discussed in this subsection.

1) Malicious Node Detection: We investigate the management of trust records by simulation that reveals important insight of the effects of various attack models. The simulation is set up as follows. Each node randomly selects one of its neighbors to transmit packets. Suppose that node *A* asks node *B* to forward packets; node *A* can observe how many packets *B* has forwarded. Next, node *A* manages its trust record $T(\text{A: B, packet forwarding})$ using the procedure described in Section IV-C. Also, node *A* can observe the leaving time (respectively, the transmission rate) of node *B*, and then manage its trust record $T(\text{A: B, leaving time})$ (respectively, $T(\text{A: B, transmission rate})$) using the procedure described in Section IV-C. At the same time, node *A* manages the trust records of two-hop neighbors by calculating the recommended trust values. In this simulation, the number of forwarded packets in each time unit [i.e., N in (1)] is set to 500. A malicious node drops the packets from the honest nodes with packet drop ratio 85%, and/or sends a large volume of bogus packets to jam the communication channels at a transmission rate $1.8 \times \theta_2$. We assume that the leaving time for a compromised node is longer than five intervals of HELLO messages and δ in (2) is ten intervals of HELLO messages. On the other hand, in each time unit, the leaving time of a legitimate node, which is observed by its one-hop neighbors, is less than three intervals of HELLO messages. Moreover, the maximum number of historical records for each behavior is 10.

Four types of malicious nodes are considered. Type 1 drops packets only, type 2 sends a large volume of bogus packets only, type 3 not only drops packets but also sends a large volume of bogus packets, and type 4 is a compromised node. We define type 0 nodes to be legitimate nodes.

In Fig. 4, we show the trust record in a legitimate node about its neighbors' transmission rates and packet forwarding at different time units. The community size of the legitimate node is 20, where 16 nodes are malicious, and 4 nodes for types 1, 2, 3, and 4, respectively. Here, S is the simulation time units, which indicates the interaction times between each node and other nodes of its community. Fig. 4(a) plots the transmission rate trust values versus packet-forwarding trust values of all one-hop neighbors. At the beginning of the simulation (e.g., $S \leq 5$), most of the nodes are with the value of 0.5 in either transmission rate trust or packet-forwarding trust. The reason is that this node does not have much experience with its neighbors. With more observations, legitimate nodes form a cluster that is close to top-right corner and this cluster becomes tighter and tighter. Three types of malicious behaviors are clearly shown and can be differentiated. Type 1 nodes are in the upper-left area, type 2 nodes are in the lower-right area, and type 3 nodes are in the lower-left area. It is demonstrated that with enough observation (e.g., $S \geq 100$), legitimate nodes and different types of malicious nodes should form clusters on this 2-D plot, which can be used to separate legitimate and malicious nodes. It is clear that after $S \geq 100$, the trust values of one-hop neighbors become

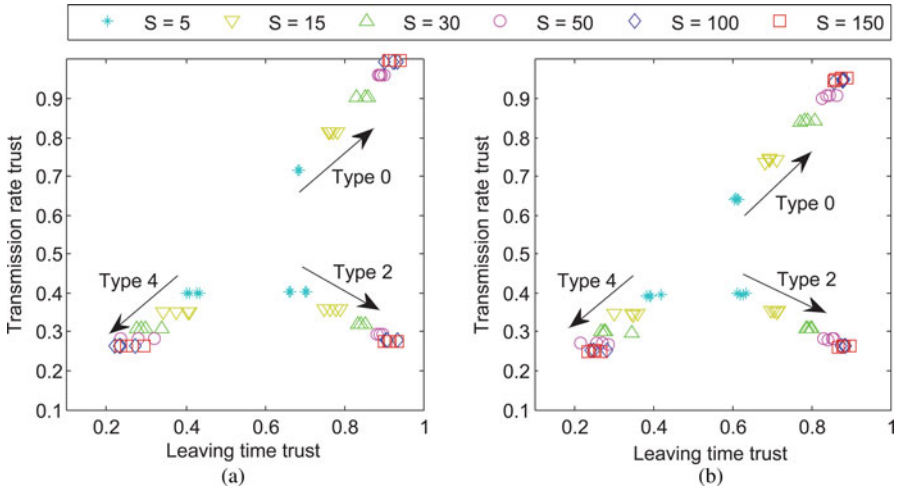


Fig. 5. Trust record in a legitimate node about its neighbors' leaving time and transmission rate. (a) One-hop neigboring nodes. (b) Two-hop neigboring nodes.

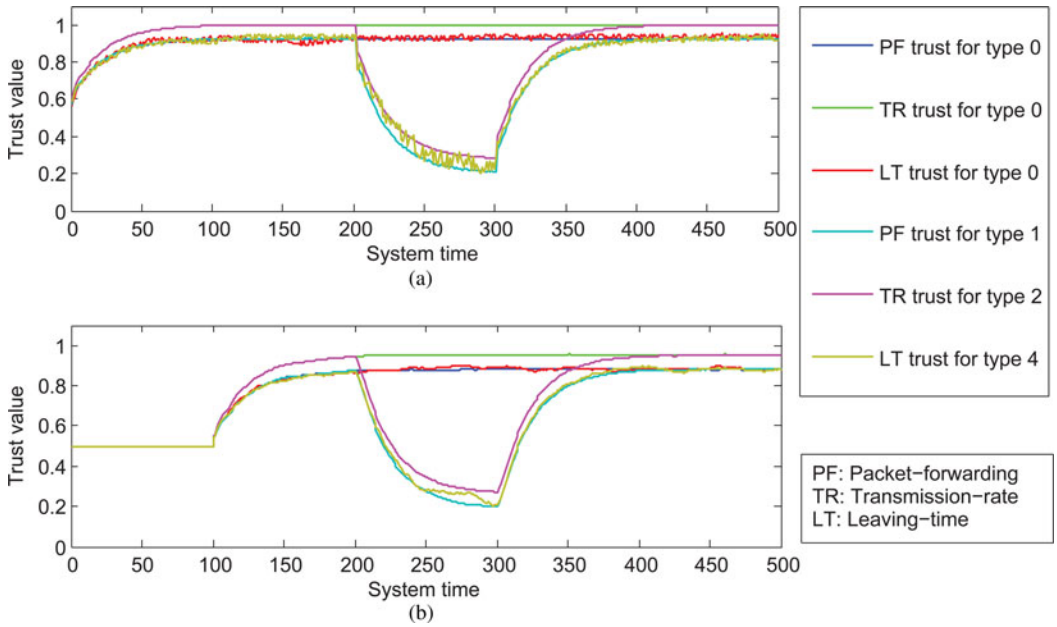


Fig. 6. Trust record in a legitimate node under one specific type of attack. (a) One-hop neigboring nodes. (b) Two-hop neigboring nodes.

stable. Fig. 4(b) shows the trust record of two-hop neighbors. In our simulation, as described in Section IV-A, the beginning time of calculating the recommended trust values of two-hop neighbors is 50 time units later than that of calculating the direct trust values of one-hop neighbors. Similar to Fig. 4(a), with more observations, legitimate nodes form a cluster that is close to top-right corner and this cluster becomes tighter and tighter. Three types of malicious behaviors are clearly shown and can be differentiated.

Fig. 5 shows the trust record in the legitimate node about neighbors' leaving time and transmission rate at different time units. Fig. 5(a) and 5(b) show the trust record of one-hop neighbors and two-hop neighbors, respectively. Obviously, the aforesaid conclusions are also applicable in this case.

Having studied the impact of multiple types of attack simultaneously, we next focus on individual attack type separately. In each case, the attack begins and stops when the system time equals to 200 time units and 300 time units, respectively. The results for individual attack type are all plotted in Fig. 6. Here, the beginning time of calculating the recommended trust values of two-hop neighbors is 100 time units later than that of calculating the direct trust values of one-hop neighbors. It is clear that the trust value of a legitimate node increases very quickly, becoming more than 0.85 after a short time (i.e., 32 time units for one-hop neighboring nodes and 45 time units for two-hop neighboring nodes), and then remaining stable. On the other hand, the trust value of a malicious node decreases very quickly, becoming less than 0.3 after a short time (i.e., 40 time units), and remaining stable. Also, once a malicious node finishes attacking (e.g., the node has been replaced), the dynamics of its trust value exhibits similar behaviours as that of a legitimate node.

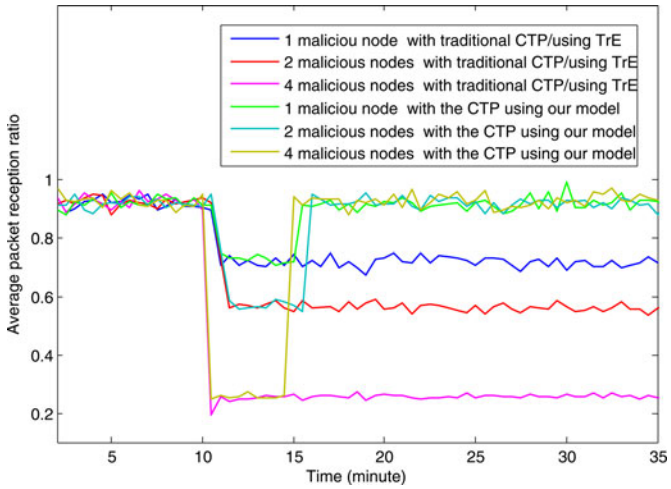


Fig. 7. Average PRR with different number of malicious nodes about packet forwarding.

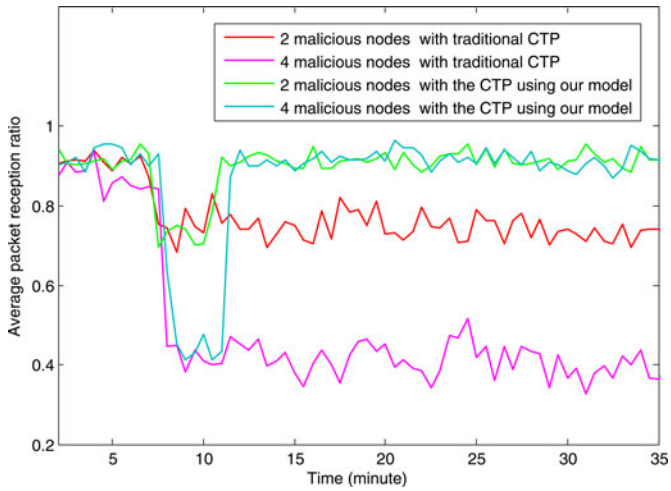


Fig. 8. Average PRR with different number of malicious nodes about DoS attacks.

2) Average PRR Improvement: We evaluate the average PRR improvement by implementing the CTP with our trust system on an experimental testbed.

Traditional CTP is a distributed routing protocol for tree-based data collection in WSNs. As the reference routing protocol for TinyOS 2.x [23], it has been strenuously tested and shown to work well in sensor networks. It can also be used by the base station in an MSN to collect medical data from each sensor node. For simplicity, here we assume that each node only manages trust records of its one-hop neighbors about performing three behaviors as mentioned in Section IV. That is, only direct trust and historical trust are used. Moreover, the maximum number of historical records for each behavior is 10, and all the aging factors are set to 0.1. We use an indoor testbed consisting of 20 TelosB motes and a base station (performed by a TelosB mote) to compare the performance of traditional CTP, the CTP using *TrE*, and the CTP using our proposed model. We deploy 20 nodes in a 4×5 grid where the separation between neighboring grid points is 0.5 m. Among the 20 nodes, the possible number of malicious nodes is 1, 2, or 4. The delivery rate of packets is based on the random sending mechanism of CTP which is about 0.67 packets per second. The data collecting nodes are located along one edge (width) of the grid, which are far from the base station. The base station calculates the average PRR every 30 seconds. Each experiment lasts about 35 min.

Fig. 7 shows the average PRR of all legitimate nodes in the case when the malicious nodes perform gray hole attack. They begin attacks on the tenth minute and randomly drop 85% packets passing through them. Traditional CTP cannot observe the packet dropping behavior of malicious nodes, although it employs the bidirectional expected transmissions (ETX) scheme. The detailed explanation is given as follows. CTP contains three main subsystems: link estimator, routing engine, and forwarding engine [23]. Routing estimation and selection mainly depend on link estimator. Link estimator uses link qualities to evaluate the neighbors based on beacons and acknowledgments (Acks). Since the malicious nodes can also echo Acks as legitimate ones, this can cause its children to choose it as their parent. Also, traditional CTP restricts the parents for deciding whether its children are legitimate ones. Hence in traditional CTP, the malicious nodes can disguise as legitimate ones to probably cause a great amount of data missing in MSNs. By introducing our proposed system, we have modified the basic CTP as follows. A trust value about performing packet forwarding is attached into each entry of the neighbor table of each sensor node. While deciding the next hop of a route, each node, say *A*, will choose a threshold value as the trust requirement of the next hop node of this route. Node *A* then checks the total trust values of all of its one-hop neighbors and selects those neighbors which meet this trust requirement. After that, *A* checks the qualified nodes' ETX and the node 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.

Fig. 7 shows that with the increase of the number of malicious nodes, the average PRR of traditional CTP falls. The average PRR of the CTP using *TrE* is the same as traditional CTP. That is because the unsuccessful packet forwarding behavior of a node is not considered in *TrE*. It is obvious that malicious nodes can significantly degrade the performance of basic CTP and the CTP using *TrE*. Even with two malicious nodes (10% of total nodes), the average PRR can be as low as 54%. Obviously, using the proposed system to build and utilize trust records can greatly improve the performance. In particular, the average PRR is more than 87% in the presence of four malicious nodes (20% of total nodes). The main difference between these three systems is that, compared to the traditional CTP and CTP using TrE, the CTP using our model efficiently avoids choosing malicious nodes as parents to forward packets to the base station.

Fig. 8 shows the average PRR of all legitimate nodes in the case when the malicious nodes perform DoS attacks. They begin attacks on the eighth minute and send one malicious packet every 1 ms. Here, λ=1. Obviously, with the increase of the number of malicious nodes, the average PRR of traditional CTP falls. Also, malicious nodes can significantly degrade the performance of

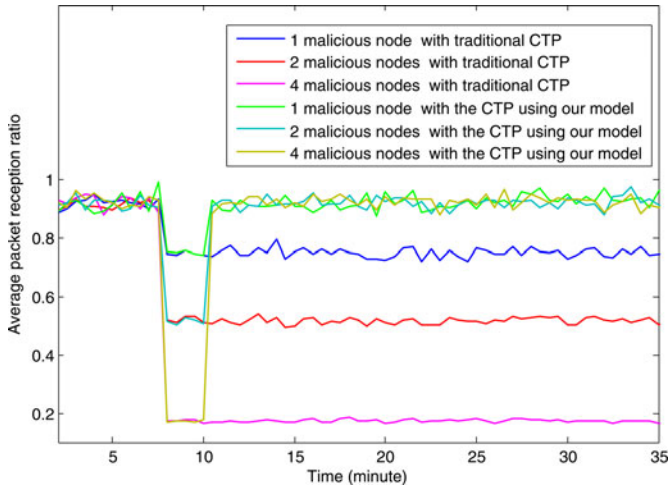


Fig. 9. Average PRR with different number of malicious nodes about leaving.

basic CTP. Even with two malicious nodes (10% of total nodes), the average PRR can be as low as 69%. Obviously, using the proposed system to manage trust records can greatly improve the performance. In particular, even if there are four malicious nodes (20% of total nodes), then the average PRR degrades less than 11%.

Fig. 9 shows the average PRR of all legitimate nodes in the case that there are compromised (malicious) nodes leaving the network. These nodes are compromised on the eighth minute. Once they are compromised, they drop 100% packets passing through them but acknowledging the packets they received. Obviously, with the increase of the number of malicious nodes, the average PRR of traditional CTP falls. Also, malicious nodes can significantly degrade the performance of basic CTP. Even with two malicious nodes (10% of total nodes), the average PRR can be as low as 49%. On the other hand, using the proposed system to manage trust records can greatly improve the performance. In particular, the average PRR is greater than 88%, even with four malicious nodes (20% of total nodes). From the results in Fig. 8 and Fig. 9, it can be seen that the main difference between these two systems is that, compared to the traditional CTP, the CTP with our model efficiently identifies the malicious nodes and then removes them. Note that since *TrE* does not consider the leaving time and transmission rate, here we do not compare it with the proposed model with respect to these two behaviors.

VI. Application Scenarios

The proposed trust evaluation model can be employed in various application scenarios (e.g., secure multicast, distributed data storage, and multimedia traffic security architecture [24]). Here, we consider centralized malicious node detection and secure unicast routing as two examples.

A. Centralized Malicious Node Detection

As described in Section IV, each sensor node can monitor the activities of others nearby. Once some abnormal activities are observed (e.g., the total trust value is lower than a predefined threshold), a node may raise an alert to the base station who further determines which nodes are compromised. Here, a node can use its individual key to encrypt an alert message and then delivers it to the base station. An alert reasoning algorithm to effectively use such alerts to identify compromised nodes has been proposed in [25]. The algorithm assumes that compromised nodes may collude at will. It has been demonstrated that the algorithm is optimal in the sense that it identifies the largest number of compromised nodes without introducing false positives.

B. Secure Unicast Routing

Securing routing is a fundamental challenge for MSN security. In wireless networks, traditional approaches to secure routing focus on preventing attackers from entering the network through secure key distribution/authentication and secure neighbor discovery, such as [26]. However, those protocols are susceptible to a variety of node misbehaviors. Therefore, it is important to develop mechanisms to monitor route disruption in MSNs and adjust route selection dynamically. Here, we use the proposed trust model to improve unicast routing.

Consider that a source node wants to communicate with the base station. There are two feasible approaches for the use of the proposed trust model. One is that the source node checks the total trust values of all of its one-hop neighbors and selects the neighbor with the highest trust value. Thus, the source node encrypts information by its individual key before sending the encrypted information to the qualified node. When the qualified neighbor receives the encrypted message, it will follow the same procedure for selecting a neighbor with the highest trust value. The other is that the source node first tries to find multiple routes to the base station. Then the source tries to find the total trust value of the nodes on the routes from its direct trust and recommended trust. Finally, the source node selects the most trustworthy route to transmit data.

VII. Conclusion

In this paper, we have presented a novel distributed trust evaluation model for MSNs, where each node manages trust records of other nodes about performing some activities. Further, we showed the usage of the proposed trust model in a variety of application scenarios in MSNs such as centralized malicious node detection and secure unicast routing. The proposed model was evaluated using both computer simulations and experimental testbeds and has shown promising results, in effectively identifying malicious nodes and improving packet delivery.

References

[1] 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.

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

[3] O. G. Morchon and H. Baldus, "Efficient distributed security for wireless medical sensor networks," in *Proc. Int. Conf. Intel. Sensors, Sensor Networks and Inf. Proces.*, pp. 249–254, 2008.

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

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

[6] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," in *Proc. SenSys*, 2009, pp. 1–14.

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

[8] B. Yu, M. Singh, and K. Sycara, "Developing trust in large-scale peer-to-peer systems," in *Proc. IEEE First Symp. Multi-Agent Security and Survivability,*, 2004, pp. 1–10.

[9] 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.

[10] S. Marti, T. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *Proc. Mobicom 2000*, pp., 255–265.

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

[12] G. Theodorakopoulos and J. S. Baras, "Trust evaluation in ad-hoc networks," in *Proc. ACM WiSec*, 2005, pp. 1–10.

[13] M. K. Denko, T. Sun, I. Woungang, J. Rodrigues, and H.-C. Chao, "A trust management scheme for enhancing security in pervasive wireless networks," in *Proc. GLOBECOM*, 2009, pp. 1–6.

[14] K. Wang, M. Wu, and S. Shen, "A trust evaluation method for node cooperation in mobile ad hoc networks," in *Proc. 5th Int. Conf. Inf. Technol: New Gen.*, 2008, pp. 1000–1005.

[15] D. Liu, P. Ning, and W. Du, "Detecting malicious beacon nodes for secure location discovery in wireless sensor networks," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst.*, 2005, pp. 609–619.

[16] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi, "The flooding time synchronization protocol," in *Proc. Sensys*, 2004, pp. 39–49.

[17] C. E. Perkins and M. E. Royer, "Ad hoc on demand distance vector (AODV) routing," (Jul. 2003). [Online]. Available: http://tools.ietf.org/html/rfc3561

[18] W. Du, J. Deng, Y. Han, P. Varshney, J. Katz, and A. Khalili, "A pairwise key predistribution scheme for wireless sensor networks," *ACM Trans. Inf. Syst. Security*, vol. 8, no. 2, pp. 228–258, 2005.

[19] S. Zhu, S. Setia, and S. Jajodia, "LEAP+: Efficient security mechanisms for large-scale distributed sensor networks," *ACM Trans. Sens. Netw.*, vol. 2, no. 4, pp. 500–528, 2006.

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

[21] H. Song, L. Xie, S. Zhu, and G. Cao, "Sensor node compromise detection: the location perspective," in *Proc. Int. Conf. Wireless Commun. and Mobie Comput.*, 2007, pp. 242–247.

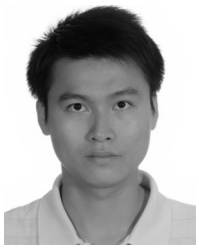
[22] A. Abidoye, N. Azeez, A. Adesina, K. Agbele, and H. Nyongesa, "Using wearable sensors for remote healthcare monitoring system," *J. Sens. Technol.*, vol. 1, no. 2, pp. 22–28, 2011.

[23] TinyOS: An open-source OS for the networked sensor regime, (Jul. 2003). [Online]. Available: http://www.tinyos.net/

[24] L. Zhou and H.-C. Chao, "Multimedia traffic security architecture for the internet of things," *IEEE Netw. Mag.*, vol. 25, no. 3, pp. 35–40, May/Jun. 2011.

[25] 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.

[26] Y.-C. Hu and A. Perrig, "A survey of secure wireless ad hoc routing," *IEEE Security Privacy*, vol. 2, no. 3, pp. 28–39, May/Jun. 2004.



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

His research interests include network and systems security. He serves on the Editorial Boards of the Wiley's Wireless Communications and Mobile Computing Journal, the Wiley's Security and Communication Networks Journal, and KSII Transactions on Internet and Information Systems. Also, he is a Technical Program Committee Member of the IEEE GLOBECOM 2011, IEEE PIMRC 2011, IEEE ICC 2012, and IEEE WCNC 2012.

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

He is a Professor in 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, Australia, in 1988 and 1990, respectively, and the Ph.D. degree in communication engineering from the Royal Melbourne Institute of Technology, Melbourne, Australia, 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, where he is currently an Associate Professor.

Jiajun Bu received the B.S. and Ph.D. degrees in computer science from Zhejiang University, Zhejiang, 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 at 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, the M.S. degree in computer engineering from the University of Massachusetts, Amherst, and the Ph.D. degree in computer engineering from the University of Patras, Patras, Greece, in 1983, 1986, and 1988.

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

Dr. Vasilakos has served as General Chair and Technical Program Committee Chair for many international conferences. He served or is serving as an Editor or/and Guest Editor for many technical journals. He is the General Chair of the Council of Computing of the European Alliances for Innovation.