PAPER

Trusted Routing Based on Dynamic Trust Mechanism in Mobile Ad-Hoc Networks

Sancheng PENG[†,††], **Weijia JIA**[†,†††*], **Guojun WANG**[††††,†††††], **Jie WU**[†††††], ***Nonmembers,***
and* Minyi GUO**[††††††], ***Member

SUMMARY Due to the distributed nature, mobile ad-hoc networks (MANETs) are vulnerable to various attacks, resulting in distrusted communications. To achieve trusted communications, it is important to build trusted routes in routing algorithms in a self-organizing and decentralized fashion. This paper proposes a trusted routing to locate and to preserve trusted routes in MANETs. Instead of using a hard security mechanism, we employ a new dynamic trust mechanism based on multiple constraints and collaborative filtering. The dynamic trust mechanism can effectively evaluate the trust and obtain the precise trust value among nodes, and can also be integrated into existing routing protocols for MANETs, such as ad hoc on-demand distance vector routing (AODV) and dynamic source routing (DSR). As an example, we present a trusted routing protocol, based on dynamic trust mechanism, by extending DSR, in which a node makes a routing decision based on the trust values on its neighboring nodes, and finally, establish a trusted route through the trust values of the nodes along the route in MANETs. The effectiveness of our approach is validated through extensive simulations.

key words: *Mobile ad-hoc networks, trust evaluation, multiple constraints, collaborative filtering, trusted routing.*

1. Introduction

Compared with traditional networks, mobile ad-hoc networks (MANETs) are vulnerable to various attacks due to their characteristics, such as dynamic network topology, restricted power supply, limited computational abilities, and continuously changing scale. Thus, in order to enhance the security of MANETs, it is important to assess the trust of nodes and design more effective routing algorithms.

In the past decade, research on the subject of trust has been extensively performed for a wide range of applications in many areas, such as e-commerce and peer-to-peer (P2P) networks. Incorporating the notion of trust into MANETs and sensor networks has recently gained a large amount of attention, including many trust models [1-6] and secure routing protocols [7-12].

Although much research has been conducted, there still exists some challenging research issues in MANETs. Generally, the following issues for trusted routing in MANETs should be considered thoroughly:

First, how to consider various factors simultaneously, such as the time of interactions (i.e., forward data packets), the number of interactions, and the amount of interactions in evaluating the trust of mobile nodes? How to ensure that trust increases slowly, while dropping fast? How to adopt incentive mechanisms, e.g., the positive behaviors can be rewarded and the negative behaviors can be punished? How to differentiate false recommendation from honest recommendation, e.g., the malicious nodes collude with each other to accuse non-malicious nodes of being bad? Second, how to design a routing algorithm by using the trust mechanism to prevent attacks from disrupting the routing services?

In this paper, we propose a trusted routing protocol based on the dynamic trust mechanism, by extending the dynamic source routing (DSR) protocol [13] in MANETs, DTM-DSR for short. We present the specific contributions towards DTM-DSR as follows:

1. We present a trust updating algorithm that guarantees the slow rise and rapid drop of trust due to these three constraints: the time aging factor can ensure that the trust fades with time; the rewards factor can ensure that the positive behavior is rewarded; and the penalty factor can ensure that the negative behavior is punished.
2. We adopt collaborative filtering [14-15] to access the recommendation trust to prevent dishonest recommendations.
3. We associate trust levels with network nodes in order to compute trusted routes through the proposed dynamic trust mechanism.
4. Through simulation studies, we compare the performance of the DTM-based DSR protocol with that of the DSR protocol based on Bayesian (Bayesian-DSR) and the traditional DSR protocol under three evaluation metrics.

The remainder of this paper is organized as follows: Section 2 gives the related work. Section 3 presents the dynamic trust evaluation mechanism. Section 4 gives DTM-DSR, which is an extension of DSR with the dynamic trust mechanism. Section 5 provides the simulation studies. Finally, we conclude this paper with directions for future research in Section 6.

Manuscript received July 01, 2009.
Manuscript revised October 08, 2009.

[††††] The authors are with the School of Information Science and Engineering, Central South University, Changsha, Hunan Province, P.R. China, 410083.

[†] School of Computer and Communication, Hunan University of Technology, Zhuzhou, Hunan Province, P.R. China, 412008.

[†] Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong.

[†††††] Department of Computer and Information Sciences at Temple University, USA.

[††††††] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, P.R. China, 200030.

[*] The corresponding author. Email: wei.jia@cityu.edu.hk

DOI: 10.1587/trans.E0.??.1

2. Related Work

In this section, we investigate related work in two dimensions. One dimension is about the trust evaluation models in MANETs, and the other is related to the trust/reputation-based routing protocols in MANETs. Many of the existing routing protocols are the extensions of the popular on-demand routing protocols, such as DSR and AODV [16].

2.1 Trust Evaluation Model

Sun et al [1] propose a trust model based on entropy. The proposed model can capture the uncertainty of the message space itself and thus it is useful for estimating the uncertainty of the trust relation. But it is not a general mathematical model, and can not prevent the false recommendations.

Theodorakopoulos et al [2] propose a semiring-based trust model. The proposed model can evaluate trust by using the path semiring and the distance semiring. It also has more dynamic adaptability, but its convergence is slow and cannot be adopted in large-scale networks.

Li et al [3] propose an objective trust management framework (OTMF) based on a modified Bayesian approach. In OTMF, the confidence value is included in trust evaluation. Trust in OTMF is formed based not only on direct observations, but also on the second-hand information. It does not consider the recommendation trust.

Anantvalee et al [4] propose a reputation-based system as an extension to source routing protocols for detecting and punishing selfish nodes in MANETs. It can encourage suspicious nodes to cooperate, using a reputation management system. But it does not consider how to compute the reputation value of nodes.

Peng et al [5] propose a trust model based on Bayesian theory. The model assesses subjective trust of nodes through the Bayesian method, which makes it easy to obtain the subjective trust value of one node on another, but it cannot detect dishonest recommendations.

Luo et al [6] propose a fuzzy trust recommendation framework, and the recommendation algorithm is based on collaborative filtering in MANETs. It considers recommendation trust, while it does not consider other factors, such as the time aging and the certainty nature of trust.

2.2 Trust/Reputation-based Routing Protocol

Pirzada et al [7] propose a dependable routing by incorporating trust and reputation in the DSR protocol. The mechanism makes use of Route Reply packets to propagate the trust information of nodes in the network. These trust values are used to construct trusted routes that pass through benevolent nodes and circumvent malicious nodes. But it does not consider how to prevent dishonest recommendation in the trust model.

Wang et al [8] propose the cooperative on-demand secure route (COSR) protocol to use against the main passive route attacks. COSR measures node-reputation (NR) and route-reputation (RR) by contribution, Capability of Forwarding (CoF), and recommendation to detect malicious nodes.

Marti et al [9] propose two techniques, Watchdog and Pathrater. The Watchdog promiscuously listens to the transmission of the next node in the path for detecting misbehaviors. The Pathrater keeps the ratings for other nodes and performs route selection by choosing routes that do not contain selfish nodes. However, the watchdog mechanism needs to maintain the state information regarding the monitored nodes and the transmitted packets, which would add a great deal of memory overhead.

Michiardi and Molva [10] propose a collaborative reputation (CORE) mechanism, which also uses the watchdog mechanism to observe neighbors, and aims to detect and isolate selfish nodes. However, only positive indirect reputation is allowed in this system to avoid false accusation and denial of service attacks.

Buchegger et al [11] propose an extension to the source routing protocol, cooperation of nodes, fairness in dynamic ad hoc networks (CONFIDANT). When misbehaving nodes are detected, it sends an alarm to other nodes in the network, defined as friends, to isolate misbehaving nodes from the network.

Yu et al [12] propose an attack detection and defense mechanism by using both the route redundancy in ad hoc networks and the message redundancy in topology discovery of the routing protocols. But it does not consider how to prevent dishonest recommendation in the trust model.

3. Dynamic Trust Mechanism

Most statistical methods assume that the behavior of a system is stationary, so the ratings can be based on all observations back to the beginning of time. But often the system's behavior changes with time, and our main interest is to reward the positive behavior and to punish the negative behavior. Thus, we present a dynamic trust mechanism (DTM) in this section. In DTM, we introduce the trust updating algorithm with multiple constraints to assess the direct trust by the self-experience of nodes to provide the assurance for the slow rise of trust, the rapid decreasing of trust (similar to the trust building in the real human society), and ensure that the trust fades with time. Moreover, we introduce collaborative filtering technique to evaluate recommendation trust in order to detect and prevent the false recommendations.

3.1 Description of Trust

Currently, the research on trust focuses on two aspects: Objective trust and subjective trust. The model of subjective trust has become a great concern on the research of trust domains. There are three definitions on trust [17] as follows:

***Definition* 1:** Trust is the subjective probability of one entity expecting that another entity performs a given action on which its welfare depends. The first entity is called *trustor*, while the other is called *trustee*.

***Definition* 2:** Direct trust refers to an entity's belief in another entity's trustworthiness within a certain direct interaction to a certain direct experience.

***Definition* 3:** Recommendation trust refers to one entity which may also believe that another entity is trustworthy due to the recommendations of other entities with respect to their evaluation results.

3.2 Trust Updating Algorithm with Multiple Constraints

We introduce a trust updating algorithm with multiple constraints, such as time aging factor, rewards factor, and penalty factor, to assess the direct trust by the self-experience of nodes. Similar to the human society, trust should fade with time. Moreover, the rewards factor and the penalty factor are used to distinguish the impact of successful and failed interactions for the evaluation of trust.

Let the trust value of each node be updated periodically within each time period T. The trust value can be calculated as follows:

$$T^d_{new}(i,j) = \begin{cases} 1 - TF \times T^d_{old}(i,j), \\ \quad \text{if } (s_{new} = 0 \text{ and } f_{new} = 0 \\ \quad\quad \text{and } T^d_{old}(i,j) > 0) \\ (1 - TF \times (RF \times S - PF \times F)) \\ \times T^d_{old}(i,j) + TF \times (RF \times S - PF \times F), \\ \quad \text{if}(\ (s_{new} > 0 \text{ or } f_{new} > 0) \\ \quad\quad \text{and } T^d_{old}(i,j) > 0) \\ 0, \text{ otherwise} \end{cases} \tag{1}$$

Where: $TF = \frac{\lambda e^{C_1 \times \Delta t} - 1}{\lambda e^{C_1 \times \Delta t} + 1}$ is a time aging factor, which represents that the trust fades with time. $RF = \frac{\lambda e^{C_2 \times s_{new}/\Delta t} - 1}{\lambda e^{C_2 \times s_{new}/\Delta t} + 1}$ is a rewards factor, which denotes the positive impact for the trust in successful interactions during the time period Δt. $PF = \frac{\lambda e^{C_3 \times f_{new}/\Delta t} - 1}{\lambda e^{C_3 \times f_{new}/\Delta t} + 1}$ is a penalty factor, which denotes the negative impact for the trust in failure interactions during the time period Δt. λ is a constant, which can be determined according to the practical requirement. C_1, C_2, and C_3 are adjusted factors for time aging, rewards, and penalty, respectively. Δt denotes the period between the current time and the time of last interaction between node i and node j ($\Delta t \geq 0$). Δt can be determined according to the practical requirement. s_{new} and f_{new} denote the amount of successful and failed interactions, respectively, during time period Δt ($s_{new} \geq 0$, $f_{new} \geq 0$). $S = s_{new}/(s_{new} + 1)$ and $F = f_{new}/(f_{new} + 1)$.

$T^d_{new}(i,j)$ denotes the new direct trust of i on j.

3.3 Recommendation Trust Based on Collaborative Filtering

To avoid malicious nodes submitting dishonest recommendations, or colluding with each other to boost their own trust, or accusing non-malicious nodes of misbehaving, we introduce the recommendation systems from [15], which enables nodes to find recommendations about the quality of information, or services. Collaborative filtering is widely adopted in the applications (e.g., a movie, news, or commodities) of recommendation systems. Through collaborative filtering, nodes can get personal predictions and suggestions to help them find what they want with a higher probability.

Thus, in our mechanism, collaborative filtering is used to compute the recommendation trust of a node. The main idea is: A node has the similar trust preferences on some nodes as the target node. Then, it provides recommendations or predictions to the target node based on the same preferences. If the similarity between a node and the target node is larger, the recommendation credibility of the node is higher.

There are four major processes in collaborative filtering: Representation, similarity decision, neighborhood information, and recommendation computation.

i) Representation: If we compute $T^r(i,j)$, let $k \in N(i)$ and $k \in N(j)$, and existing $T^d(k,j)$, then $\forall u \in N(i) \cap N(j)$, we can obtain $T^d(i,u)$ and $T^d(k,u)$.

ii) Similarity decision: We can determine the similarity between nodes i and k according to Formula (2).

iii) Neighborhood information: We find the set of K most similar nearest-neighbors, and all the similarities between node i and its neighbor nodes are computed. The K most similar nearest-neighbors are sorted by similarity.

iv) Recommendation computation: We can obtain the recommendation trust by Formula (3).

The similarity between nodes i and k, denoted by $S(i,k)$, is given as follows:

$$\begin{aligned} S(i,k) = {} & \textstyle\sum_{u \in CN(i,k)} (T^d(i,u) - \overline{T_i}) \times (T^d(k,u) - \overline{T_k}) \\ & /(\sqrt{\textstyle\sum_{u \in CN(i,k)} (T^d(i,u) - \overline{T_i})^2} \\ & \times \sqrt{\textstyle\sum_{u \in CN(i,k)} (T^d(k,u) - \overline{T_k})^2}) \end{aligned} \tag{2}$$

Where: $CN(i,k)$ denotes the number of common neighbor nodes for nodes i and k. $T^d(i,u)$ denotes the direct trust value of node i put on $CN(i,k)$. $T^d(k,u)$ denotes the direct trust value of node k put on $CN(i,k)$. $\overline{T_i}$ and $\overline{T_k}$ denote the average direct trust value of node i and node k that are put on their common neighbor nodes in $CN(i,k)$, respectively.

This function is local (1-hop neighbor nodes) and is evaluated on the recent past behaviors of node i and node k. It is locally used to prevent a false recommendation from being propagated within the network.

Recommendation trust is computed as follows:

$$T^r(i,j) = \frac{\sum_{k \in K} T^d(k,j) \times S(i,k)}{\sum_{k \in K} S(i,k)} \tag{3}$$

Where: $T^r(i,j)$ denotes the recommendation trust of node i on node j.

3.4 Evaluation of Trust

According to Formulae (1) and (3), we can obtain the trust

$T(i, j)$ between nodes i and j by combining $T^d(i, j)$ with $T^r(i, j)$ as follows:

$$T(i, j) = \omega_1 T^d(i, j) + \omega_2 T^r(i, j) \tag{4}$$

Where: $\omega_1 + \omega_2 = 1$, and ω_1 and ω_2 denote the weighting factors for $T^d(i, j)$ and $T^r(i, j)$, respectively. For example, if we think that the direct trust of one node is more trustful than the recommendation trust from other nodes, we will adopt $\omega_1 > \omega_2$.

3.5 Trust Maintenance

MANETs can not adopt an agency of trust to manage and maintain the trust of mobile nodes, but utilize the mechanism by which each node manages and maintains directly the trust value on its neighbor nodes. The mechanism has many advantages as follows: i) the overhead of management and maintenance is low; ii) it can update the trust value in real time; and iii) it can improve security to prevent malicious nodes from attacking other nodes.

However, nodes may be mobile, which causes the network topology to be dynamic. In our mechanism, each node, say i, maintains a set of neighbors $N(i)$. Let the updating time period be Δt, and N be the total number of mobile nodes. To cope with the dynamic topology changes, for each node i, there are three cases to be considered: i) a neighbor u of i moves but still in $N(i)$; ii) a neighbor u of i moves out of $N(i)$; and iii) a node v moves in and becomes the new neighbor of i. We discuss the above cases in detail as follows:

Case 1: If a neighbor u of i moves but is still in $N(i)$, we assume that the location of a neighbor u changes but u is still in $N(i)$. There are two sub-cases: a) If the set of recommendation nodes do not change, we need to update $T^d(i, u)$ and $T^r(i, u)$; b) If the set of recommendation nodes change, we need to update not only the set of recommendation nodes, but also $T^d(i, u)$ and $T^r(i, u)$.

Case 2: If a neighbor u of i moves out of $N(i)$, we only need to update $N(i)$.

Case 3: If a node v moves in and becomes the new neighbor of i, we assume that a node v moves into the coverage area of i, and becomes the new neighbor of i. We need to update $N(i)$ and set $T(i, v)$=0.5 for node i. As for node v, we need to update $N(v)$ and set $T(v, w)$=0.5 (let $w \notin N(v)$).

4. DTM-DSR

In this section, we extend the DSR protocol to which can establish trusted route based on dynamic trust mechanism, denoted by DTM-DSR. The differences between DSR and DTM-DSR are listed as follows. In DTM-DSR, i) we append the model of trust computation and fields which include s, f, and $T(i, j)$ in the neighbor table of each node; ii) we append two fields in the route reply message (RREQ), which includes T_{low} and BlackList. T_{low} denotes the threshold of trusted node; BlackList denotes distrusted node list; iii) we append T_{route} field in the route reply message (RREP), and T_{route} denotes the accumulated route trust.

To facilitate the analysis, we make the following assumptions: i) Each node has the same transmission radius; and ii) Each node knows the IDs of its neighbor nodes by exchanging their control information.

4.1 Route Discovery

During the process of route discovery, when node A chooses another node B to forward a packet, A may suffer some attacks from B, such as black hole attack, wormhole attack, and DoS attack. Thus, a reliable relationship between A and B should be established. A trusted route represents a route that only involves trustworthy nodes. Sending packets by the trusted route will decrease the probability of malicious attacks and improve the survivability of MANETs. We evaluate the trustworthiness of a route by the trust value of nodes along the route, denoted by T_{route}.

$$T_{route} = \prod_{i,j \in route} T(i, j) \tag{5}$$

In our trusted routing mechanism, the route discovery includes three processes: i) RREQ delivery; ii) RREP delivery; and iii) route selection.

4.1.1 RREQ delivery

When the source node S needs to send data to the destination node D, it first checks whether there is a feasible path found between S and D. If so, S sends the data to D; otherwise, S will start a route discovery. First, S appends its ID into the route record, and checks whether the trust on its neighbor nodes is lower than T_{low}. If so, S appends the ID of neighbor nodes into BlackList. Then, S broadcasts the RREQ packets with T_{low} and BlackList, and sets a timer window t_S, at the same time.

When any intermediate node receives a RREQ packet, it processes the request according to the following steps:

Step 1: If the pair ⟨Source ID, request ID⟩ for this RREQ packet is found in this node's list of recently seen requests, then it discards the RREQ packet and does not process it further.

Step 2: Otherwise, if this node's address is already listed in the route record in the request, then it discards the RREQ packet and does not process it further.

Step 3: Otherwise, if the target of the request matches this node's own address, then the route record in the packet contains the route by which the request this node from the source node of the RREQ packet. Intermediate node returns a copy of this route in a RREP packet to the source node.

Step 4: Otherwise, it appends its own address to the route record in the RREQ packet, and checks whether the trust on its neighbor nodes is lower than T_{low}. If it is, it appends the ID of the neighbor nodes into BlackList.

Step 5: Re-broadcast the request to the neighbor nodes.

The pseudo code of the RREQ delivery algorithm is shown in Fig. 1.

```
RREQ_Delivery ( )
{   To source node:
    if there is a feasible path found between $S$ and $D$, then
        $S$ sends data to $D$;
    else
    {   sets the route trust requirement $T_{low}$;
        sets the BlackList;
        sets a timer window $t_S$;
    }
    broadcasts RREQ with $T_{low}$ and BlackList;
    To an intermediate node:
    checks whether receive the RREQ;
    if receives the RREQ, then
        discards RREQ;
    else
    {   checks $T(i, j) > T_{low}$
        if $T(i, j) < T_{low}$, then
            appends the ID node $j$ into the BlackList;
        appends its node ID into the path queue;
        broadcasts RREQ;
    }
    To destination node:
    calls the process of route reply;
}
```

Fig. 1 The RREQ delivery algorithm

```
RREP_Delivery ( )
{
    To destination node:
    sets $T_{route}$=1;
    if receives the first RREQ, then
        sets a timer window $t_D$;
    checks the BlackList and $t_D$;
    if $t_D$ expires, then
        discards the follow-up RREQ;
    else if BlackList = = NULL, then
        sends RREP with $T_{route}$ along the path to the next hop.
    To an intermediate node:
    updates $T_{route}$ according to Formula (5);
    forwards the RREP;
    To source node:
    updates $T_{route}$ according to Formula (5);
    calls route selection;
}
```

Fig. 2 The RREP delivery algorithm

4.1.2 RREP delivery

When the destination node receives the first RREQ packet, it sets a timer window t_D. If t_D expires, it discards the follow-up RREQ packet. Otherwise, it checks whether the BlackList is empty. If not, it discards the RREQ packet; otherwise, it sets T_{route} =1, and then unicasts the RREP packet with T_{route} to the intermediate node. After receiving a RREP packet, the intermediate node computes T_{route} according to Formula (5), and updates the field of T_{route}, then it forwards the RREP packet with T_{route}. The pseudo code of RREP delivery algorithm is shown in Fig. 2.

4.1.3 Route selection

When S receives the RREP packet, if the timer window t_S does not expire, it needs to update the T_{route} field of this message. Otherwise, S discards follow-up RREP packets and picks a path with the largest T_{route} and less hops. The pseudo code of route selection algorithm is shown in Fig. 3.

```
Route_Selection( )
{
    when source node receives the RREP, then
    checks the $t_S$;
    if $t_S$ does not expire, then
        updates the $T_{route}$;
    else
    {   discards the follow-up RREP;
        selects the route with the largest $T_{route}$ and less hops;
    }
}
```

Fig. 3 The route selection algorithm

4.2 Route Maintenance

After each successful route discovery takes place, S can deliver its data to D through a route. However, the route may break at any time instant due to the mobility of nodes, or attacks. In order to maintain a stable, reliable, and secure network connection, route maintenance is necessary to ensure the system survivability. Route maintenance can be performed when all routes fail or when the timer window t_R for routing expires. Corresponding route maintenances of two cases are discussed as follows:

i) When T_R expires, if S finds that the end-to-end trust is below T^*, which is called the end-to-end trust requirement, S will select additional paths based on the history information rather than execute a route discovery again. During the transmission, if S finds the trust of a route has decreased, it sends a route check message along the route to check the route status, and sets a timeout period to wait for the route check messages from D. When S receives the reply, it will compute the route end-to-end trust again, then checks whether the route meets T^*. If the requirement can be satisfied, S will update the route cache and use the route for data transmission. If the validation is unsuccessful, route discovery is triggered.

ii) If all routes are broken, S simply initiates a new route discovery without any examination.

4.3 An Example

The mode of operation is illustrated in Figs. 4-6. To facilitate the analysis, we make the following assumptions: i) the timer window t_S and t_D do not expire; and ii) the subjective trust connections among nodes as a directed graph $G(t) = (V, E(t))$, called the trust graph (see Fig. 4). A directed arc from node i to node j, denoted by $T(i, j)$, corresponds to the trust relation that node i (referred to as *trustor*), and each arc comes with a weight interaction.

For example, if S is trying to discover a route to node D, it broadcasts a RREQ packet with T_{low}= 0.55 and BlackList = { }. After receiving the packet, each of intermediate nodes (a, c, e) append its ID to the packet, and check whether the trust on its neighbor nodes is lower than T_{low}, respectively. If so, it appends the IDs of the neighbor nodes into BlackList (see Fig. 5). Then, they re-broadcast the RREQ packet. Similarly, when intermediate nodes (b, f)

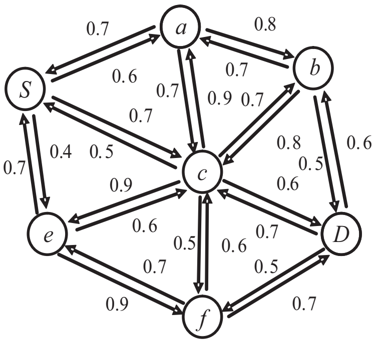

Fig. 4 Trust graph

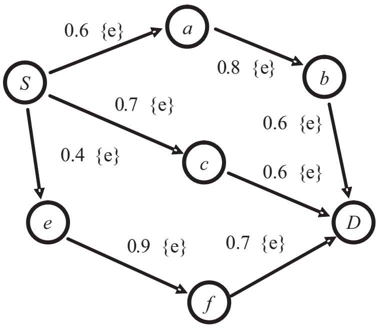

Fig. 5 RREQ delivery

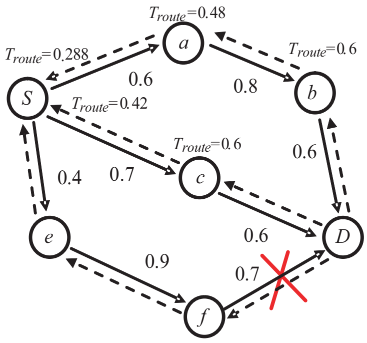

Fig. 6 RREP delivery

receive the packet, they repeat the same operation.

When D receives the RREQ packet, D checks whether the BlackList is empty. If not, it discards the RREQ packet; otherwise, the RREQ packet is replied independently through a RREP packet with T_{route}. D sets T_{route} =1, and then unicasts the RREP packet to the intermediate nodes (b, c, f). After receiving the packet, each of the intermediate nodes updates its T_{route} by the trust value on node D according to Formula (5) to its RREP packet, and then unicasts it, respectively. Similarly, when intermediate nodes (a, e) receive the RREP packets, intermediate nodes (a, e) update T_{route} through the trust value on the sending nodes (b, f), respectively, and unicast the packets (see Fig. 6)).

When the RREP packets reach S, they contain the complete trust value for the routes $S \rightarrow a \rightarrow b \rightarrow D$ and $S \rightarrow c \rightarrow D$, respectively. This information along with the existing trust information helps to select a particular route from multiple available routes. S picks the route $S \rightarrow c \rightarrow D$ according to the route selection algorithm.

Table 1 Simulation parameters

Para.	*Meaning*	*Value*
Ω	communication domain	1,000m × 1,000m
N	number of nodes	50
r	transmission radius	250m
S	maximum node speed	20m/s
P	data payload size	512 bytes/packet
w_1	weighting factor of $T^d(i, j)$	0.7
w_2	weighting factor of $T^r(i, j)$	0.3
C_1	adjusted factor for time aging	0.4
C_2	adjusted factor for rewards	0.5
C_3	adjusted factor for penalty	0.6
λ	constant	1
Δt	time interval	0.5s
T	simulation time	500 seconds

5. Simulation Studies

5.1 Simulation Parameters

To evaluate the performance of DTM-DSR, we use the simulation tool GloMoSim 2.03 [18]. We use the random waypoint as the mobility model. In our simulation, each node, at first is randomly placed in a specific field, waits for the pause time (0 second to 100 seconds), then moves to another random position with a speed chosen between 0 to 20 m/s. Every 100 seconds during the simulation, five new source and destination pairs are randomly selected, therefore, every node has a chance to be a source or a destination. The constant bit rate (CBR) is selected as the traffic model with a rate of 4 packets per second. There are three simulations carried out in this paper. Each simulation is done in the presence of five malicious nodes. The malicious nodes randomly drop data packets, with a dropping ratio in the range of 10%-30%. The related parameters are listed in Table 1.

5.2 Performance Metrics

To measure the performance of our proposed DTM-DSR, we identify three metrics: i) Throughput: The number of packets transmitted per unit time from the source node to the destination node; ii) Packet loss ratio: The ratio of the number of packets dropped to the total number of packets; and iii) Average end-to-end delay: The average delay between the sending of the packets by the source node and its receipt at the destination node.

5.3 Simulation Results

We compare the performance of DTM-DSR, Bayesian-DSR, and DSR under three metrics, including throughput, packet loss ratio, and average end-to-end delay.

Fig. 7 shows the throughput of DTM-DSR, Bayesian-DSR, and DSR. Once the packets leave the sender nodes,

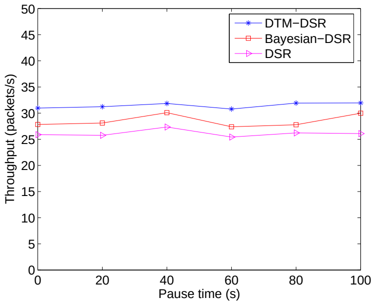

Fig. 7 Throughput of DTM-DSR, Bayesian-DSR, and DSR at different pause times

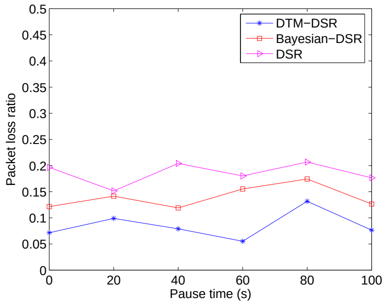

Fig. 8 Packet loss ratio of DTM-DSR, Bayesian-DSR, and DSR at different pause times

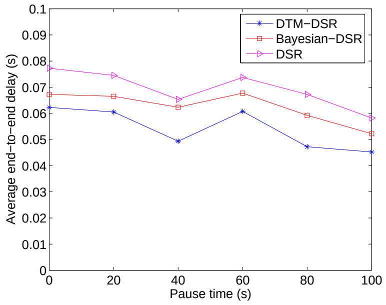

Fig. 9 Average end-to-end delay of DTM-DSR, Bayesian-DSR, and DSR at different pause times

the chances that they will be received at the destination nodes are increased, when DTM-DSR and Bayesian-DSR are used. But, the routing throughput of DTM-DSR is higher than that of Bayesian-DSR. The reason is that the DTM can evaluate the trust of nodes more precisely and avoid routing through malicious nodes more effectively.

Fig. 8 shows the packet loss ratio of DTM-DSR, Bayesian-DSR, and DSR with different pause times. It can be observed that DTM-DSR outperforms Bayesian-DSR and DSR in the packet loss ratio. The reason is that DTM-DSR always chooses a more reliable route by avoiding malicious nodes. Thus, the number of packets dropped by DTM-DSR is lower than that of Bayesian-DSR and DSR.

Fig. 9 shows the average end-to-end delay of three protocols. We can see that DTM-DSR has a smaller average end-to-end delay than Bayesian-DSR and DSR in different pause times with five existing malicious nodes. The reason is that DTM-DSR can detect malicious nodes, and thus, exclude them from routing. For DSR, if a node on an established route becomes malicious and drops packets, a connection will be timed out, and a new route discovery needs to be initiated to reestablish the route, which increases not only the total overhead, but also the average end-to-end delay; and for Bayesian-DSR, it cannot more precisely evaluate the trust of nodes, and cannot more effectively prevent dishonest recommendations. Thus, Bayesian-DSR cannot effectively avoid routing through malicious nodes.

6. Conclusions

This paper presents a protocol by extending DSR to find a trusted route in mobile ad-hoc networks. The proposed protocol is capable of identifying trustworthy nodes by using the dynamic trust mechanism under the presence of selfish or malicious nodes.

The simulation results showed the effectiveness and the superiority of DTM-DSR in the presence of selfish or malicious nodes over protocols, such as DSR and Bayesian-DSR. The proposed mechanism can also be integrated into existing routing protocols in MANETs, such as AODV.

There are still several open issues for future work, e.g., how to detect and defend simultaneously internal attacks and external attacks against routing protocols, how to quantify and evaluate the tradeoff between the security and the performance requirements of a system, and how to effectively recognize the dishonest recommendations.

Acknowledgments

This work is supported by the Research Grants Council of the Hong Kong SAR, China No. (CityU 114908), CityU Applied R & D Funding (ARD-(Ctr)) Nos. 9681001 and 9678002, the National Natural Science Foundation of China under Grant Nos. 90718034, 60740440032, 605333040 and 60811130528, and the Program for New Century Excellent Talents in University under Grant No. NCET-06-0686.

References

[1] Y. Sun, W. Yu, Z. Han, and K. J. R. Liu, “Information Theoretic Framework of Trust Modeling and Evaluation for Ad Hoc Networks,” IEEE Journal on Selected Areas in Communications, Vol. 24, pp. 305-317, 2006.

[2] G. Theodorakopoulos and J. S. Baras, “On Trust Models and Trust Evaluation Metrics for Ad Hoc Networks,” IEEE Journal on Selected Areas in Communications, Vol. 24, Issue 2, pp. 318-328, Feb. 2006.

[3] J. Li, R. Li, J. Kato, “Future Trust Management Framework for Mobile Ad Hoc Networks,” IEEE Communications Magazine, Vol. 46, Issue 4, pp.108-114, April 2008.

[4] T. Anantvalee and J. Wu, “Reputation-Based System for Encouraging the Cooperation of Nodes in Mobile Ad Hoc Networks,” Proc. of IEEE International Conference on the Communications (ICC 2007),

pp. 3383-3388, June 2007.

[5] S. Peng, W. Jia, and G. Wang, "Voting-Based Clustering Algorithm with Subjective Trust and Stability in Mobile Ad-Hoc Networks," Proc. of the IEEE/IFIP International Conference on Embedded and Ubiquitous Computing (EUC 2008), Vol. 2, pp. 3-9, Dec. 2008.

[6] J. Luo, et al., "Fuzzy Trust Recommendation Based on Collaborative Filtering for Mobile Ad-hoc Networks," Proc. of the 33rd IEEE Conference on Local Computer Networks (LCN 2008), pp. 305-311, Oct. 2008.

[7] A. A. Pirzada, A. Datta, and C. McDonald, "Incorporating trust and reputation in the DSR protocol for dependable routing," Elsevier of Computer Communications, Vol. 29, pp. 2806-2821, 2006.

[8] F. Wang, Y. Mo, B. Huang, "COSR: Cooperative On-Demand Secure Route Protocol in MANET," International Symposium on Communications and Information Technologies (ISCIT 2006), pp. 890-893, Oct. 2006.

[9] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," Proc. of MobiCom, Boston, MA, pp. 255-265, August 2000.

[10] P. Michiardi and R. Molva, "Core: A Collaborative Reputation mechanism to enforce node cooperation in Mobile Ad Hoc Networks," Communication and Multimedia Security Conference, September 2002.

[11] S. Buchegger and J.-Y. Le Boudec, "Performance Analysis of the CONFIDANT Protocol: Cooperation Of Nodes - Fairness In Dynamic Ad-hoc Networks," Proceedings of MobiHoc, June 2002.

[12] M. Yu, M. Zhou, and W. Su, "A Secure Routing Protocol Against Byzantine Attacks for MANETs in Adversarial Environments," IEEE Transactions on Vehicular Technology Vol. 58, No. 1, pp. 449-460, Jan. 2009.

[13] B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," Mobile Computing, edited by Tomasz Imielinski and Hank Korth, Chapter 5, Kluwer Academic Publishers, pp. 153-181, 1996.

[14] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-Based Collaborative Filtering Recommendation Algorithms," Proc. of the 10th International World Wide Web Conference, Hong Kong, pp. 285-295, May 2001.

[15] J. L. Herlocker, J. A. Konstan, et al., "Evaluating Collaborative Filtering Recommender Systems," ACM Trans. Information Systems, Vol. 22, No. 1, pp. 5-53, 2004.

[16] C. E. Perkins and E. M. Royer, "The Ad Hoc On-demand Distance-Vector Protocol," in Ad Hoc Networking, C. E. Perkins, Ed. Reading, MA: Addison-Wesley, Ch. 6, pp. 173-220, 2001.

[17] V. Balakrishnan, V. Varadharajan, and U. Tupakula, "Trust Management in Mobile Ad Hoc Networks," in S. Misra, I. Woungang, and S. C. Misra, Guide to Wireless Ad Hoc Networks, Springer, ISBN: 9781848003286, 2009.

[18] [Online]. http://pcl.cs.ucla.edu/projects/glomosim/.

Sancheng Peng received M.S. degree in computer science from Yanshan University. He is a student member of China Computer Federation (CCF). He is currently a Ph.D. candidate of Central South University. He is also a Research Associate at City University of Hong Kong during 2008-2009. His current research interests include computer networks, survivability, and trusted computing.

Weijia Jia is currently a full Professor in the Department of Computer Science and the Director of Future Networking Center, ShenZhen Research Institute of City University of Hong Kong. He received BSc and MSc from Central South University, China, and PhD from Polytechnic Faculty of Mons, Belgium, all in Computer Science. His research interests include next generation wireless communication, protocols and heterogeneous networks, distributed systems, multicast and anycast QoS routing protocols. He is now Chair Professor of Central South University, Changsha, China. He has served as the editor and guest editor for international journals. He is a Senior Member of IEEE and Member of ACM.

Guojun Wang received B.Sc. in Geophysics, M.Sc. in Computer Science, and Ph.D. in Computer Science, from Central South University, China. Since 2005 he is a Professor in Central South University. He is the Director of Trusted Computing Institute of the University. He is currently an Adjunct Professor at Temple University, USA. He has been a Visiting Scholar at Florida Atlantic University, USA, a Visiting Researcher at the University of Aizu, Japan, and a Research Fellow at the Hong Kong Polytechnic University, HK. His research interests include trusted computing, mobile computing, pervasive computing, and software engineering.

Jie Wu is Chair and Professor at the Department of Computer and Information Sciences at Temple University. He is an IEEE Fellow. He is on the editorial board of IEEE Transactions on Mobile Computing. He was a Distinguished Professor in Department of Computer Science and Engineering, Florida Atlantic University. He served as a Program Director at US NSF from 2006 to 2008. He has been on the editorial board of IEEE Transactions on Parallel and Distributed Systems. He has served as a distinguished visitor of the IEEE Computer Society and is the chairman of the IEEE Technical Committee on Distributed Processing (TCDP). His research interests include wireless networks and mobile computing and wireless networks, parallel and distributed systems, and fault-tolerant systems.

Minyi Guo received Ph.D. in Computer Science from the University of Tsukuba, Japan. Before 2000, he had been a Research Scientist of NEC Corp., Japan. He is now Chair and Distinguished Professor of Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. He was a Professor at the School of Computer Science and Engineering, University of Aizu, Japan. His research interests include parallel and distributed processing, parallelizing compilers, pervasive computing, embedded software optimization, molecular computing, and software engineering. He is a senior member of IEEE, a member of the ACM, IPSJ and IEICE.