Published in IET Information Security
Received on 1st August 2009
Revised on 5th April 2010
doi: 10.1049/iet-ifs.2009.0140

Special Issue on Multi-Agent & Distributed Information Security

ISSN 1751-8709

Trust-based on-demand multipath routing in mobile *ad hoc* networks

X. Li Z. Jia P. Zhang R. Zhang H. Wang
School of Computer Science and Technology, Shandong University, High-tech Development-Zone, Ji'nan 250101, Shandong Province, People's Republic of China
E-mail: lx@sdu.edu.cn

Abstract: A mobile *ad hoc* network (MANET) is a self-organised system comprised of mobile wireless nodes. All nodes act as both communicators and routers. Owing to multi-hop routing and absence of centralised administration in open environment, MANETs are vulnerable to attacks by malicious nodes. In order to decrease the hazards from malicious nodes, the authors incorporate the concept of trust to MANETs and build a simple trust model to evaluate neighbours' behaviours – forwarding packets. Extended from the *ad hoc* on-demand distance vector (AODV) routing protocol and the *ad hoc* on-demand multipath distance vector (AOMDV) routing protocol, a trust-based reactive multipath routing protocol, *ad hoc* on-demand trusted-path distance vector (AOTDV), is proposed for MANETs. This protocol is able to discover multiple loop-free paths as candidates in one route discovery. These paths are evaluated by two aspects: hop counts and trust values. This two-dimensional evaluation provides a flexible and feasible approach to choose the shortest path from the candidates that meet the requirements of data packets for dependability or trust. Furthermore, the authors give a routing example in details to describe the procedures of route discovery and the differences among AODV, AOMDV and AOTDV. Several experiments have been conducted to compare these protocols and the results show that AOTDV improves packet delivery ratio and mitigates the impairment from black hole, grey hole and modification attacks.

1 Introduction

A multi-agent system (MAS) is a system composed of multiple interacting intelligent agents. One agent communicates and cooperates with other agents to achieve its goals. A mobile *ad hoc* network (MANET) consists of wireless mobile nodes dynamically forming a self-organised MAS in which no predefined infrastructures exist and all nodes work as both host and routers. Inter-agent communication in a MANET can be achieved by message transmission. Two nodes out of direct communication range need intermediate nodes to forward their messages. MANETs are collaborative in the sense that each node is assumed to relay packets for other nodes. A MANET can exist and work well only if nodes in the network behave cooperatively. However, because of open working environment, MANETs often suffer from attacks by selfish or malicious nodes, such as packet dropping (black-hole) attacks and selective forwarding (grey-hole) attacks. As careful selection of a dependable path may mitigate the impairment from malicious nodes, how to design a dependable routing protocol is a significant problem for a MANET.

Traditional security mechanisms rely on either authenticated identities of requesting principals or the form of credentials that authorise clients to perform certain actions [1]. Generally speaking, these mechanisms rely on some underlying infrastructures, for example, public key infrastructure. However, the nature of MANETs is free of fixed infrastructures, so these mechanisms do not fit a MANET. With authentication and encryption mechanisms, secure routing protocols [2–4] have been developed to ensure properties such as confidentiality and integrity. These protocols require a centralised trusted third party, which is impractical for MANETs [5]. In addition, secure routing protocols usually cannot prevent malicious or compromised nodes from doing misbehaviours, which are authorised as participants to the network. Similar to human society where one person trusts another to carry out an action [6], the concept of trust can be introduced into

MANETs to measure an expectation or uncertainty that an entity has about another's future behaviours. It is clear that trust relationship involves two entities (subject and object) and a specific action. The uncertainty of trust exists because subject is not sure whether the object will carry out the action or not. The trust-based routing protocols are not absolutely secure but certainly have an accurate measure of reliability in them [7].

There are two primary motivations associated with trust management in MANETs. Firstly, trust evaluation helps identify malicious entities. One entity can remember others' behaviours through evaluation history. This memory provides a method for good entities to avoid working with 'ex-convict' or suspected ones. Secondly, trust management offers a prediction of one's future behaviours and improves network performance. The results of evaluation can be directly applied as an incentive for a good or honest behaviour while a penalty for a selfish or malicious behaviour in the network. The feedback reminds network participants to act with caution.

In this paper, we introduce a simple trust model based on packet forwarding ratio to evaluate neighbours' behaviours. In the model, a node trust is represented as a weighted sum of forwarding ratio of packets and a continued product of node trusts is computed as path trust. Then we propose a novel multipath reactive routing protocol for MANETs, termed as *ad hoc* on-demand trusted-path distance vector (AOTDV). In this protocol, a source can establish multiple loop-free paths to a destination in one route discovery process. Each path has an evaluation vector composed of a hop count and a trust value. A destination will respond with at most k shortest paths as candidates that satisfy the trust requirements of data packets. The shortest one will be selected as the forwarding route. As an intelligent agent, each node evaluates its neighbours' behaviours and selects the shortest trusted path to forward packets. To compare the procedures of route discovery in AODV, AOMDV and AOTDV, we depict a routing example in details and perform some experiments. The experiment results show that AOTDV improves packet delivery ratio and decreases the average response time. As a trusted multipath routing protocol, AOTDV enhances the dependability of forwarding packets and alleviates the threats from malicious nodes.

Unlike existing security mechanisms based on public key infrastructure, AOTDV routing does not rely on a trusted third party to certify keys or sell/redeem tokens. Instead, AOTDV only uses forwarding ratio of packets to recognise node behaviours. The proposed routing protocol is practical to enhance the dependability of communication and detect malicious nodes in MASs. In particular, the main contributions of our work can be summarised as follows:

1. An on-demand multipath routing protocol (AOTDV) is proposed for MANETs, in which the first k shortest trusted paths are computed out as candidates during one route discovery.

2. The traditional routing protocols only care about the number of hops whereas the simple trusted protocol (ST-AODV [5]) uses node trust values of next-hops to make routing choices. We consider the trust values of paths to the destination as well as the number of hops, so that the next hop selected indicates the shortest trusted path.

3. During the procedure of forwarding packets, QoS-aware path selection in candidates is established to satisfy user-specific requirements for dependability. An adaptive hop-by-hop mechanism is proposed to select a forward path dynamically in terms of the trust requirement posed by a packet.

4. We evaluate AOTDV protocol using the NS-2 simulator and the experimental results prove that our trust model is effective. AOTDV protocol performs better than AODV and AOMDV as it gives higher delivery ratio and lower end-to-end latency with the help of the trust model.

The rest of the paper is organised as follows. Section 2 discusses the related work. Section 3 gives a simple trust model. Section 4 describes a trust-based on-demand routing protocol. Simulation results are shown in Section 5, followed by discussion in Section 6. Conclusions are drawn in Section 7.

2 Related work

Although there has been substantial work on trust models, their applicability in mobile agent systems has received limited research attention. Trust-based routing protocols combine ideas from two research fields – trust models in trust management and routing protocols for MANETs.

2.1 Trust models

One of the earliest literatures about computational trust is Marsh's formalism [8] that uses the outcomes of direct interactions among entities to compute situational and general trust. Situational trust is the level of trust in another entity for a specific situation, whereas general trust means overall trustworthiness in spite of the situation.

Several trust models have been developed for trust management. These models can be classified into two groups: centralised models and decentralised models.

In centralised models, trust values are maintained by a common central node or an authorised third party. The simplest method is to keep a record which is equal to the number of positive ratings minus that of negative ones. This method is used in eBay's reputation forum [9]. The requirement of a trusted third party goes against the nature of MANETs.

In decentralised models, each node assigns and keeps trust/trustworthiness values for other communicators. Most researchers [10–15] are advocating the use of ratings and prefer making use of rating aggregation algorithms to evaluate the trust from several aspects (e.g. CPU usage, residual energy, bandwidth etc.). However, these sophisticated models are not appropriate for MANETs where resources are limited and network topology is dynamic. Several trust models [16–19] have been developed for peer-to-peer systems based on shared recommendation information to establish reputation. Although these models, in principle, can be applied to routing in MANETs, recommendation information exchange will incur significant network traffic. In particular, Pirzada and McDonald [7] proposed an aggregation mechanism, where nodes calculate trust according to multiple observed events including acknowledgements, packet precision, gratuitous route replies and blacklists. They have obtained promising simulation results, but we argue that similar promising effects can be obtained with a simplified trust model.

2.2 Routing protocols

Traditional routing protocols in *ad hoc* networks can be categorised into two primary types: proactive and reactive [20]. Proactive routing protocols establish and maintain routes all the time in order to avoid the latency during new route discoveries. Reactive routing protocols discover routes only when one node tries to transmit packets to another unknown-route node so as to save resources. The nodes in an *ad hoc* network generally have limited resources, such as bandwidth and power energy; therefore reactive routing protocols attract more interest. Perkins *et al.* [21] proposed a reactive single path routing protocol AODV which combines the destination sequence number in DSDV [22, 23] with the on-demand route discovery technique in DSR [24]. Unlike DSR which uses source routing, AODV is based on a hop-by-hop routing mechanism. Extended from AODV, AOMDV [25] discovers multiple loop-free and link-disjoint paths. Experiments show that AOMDV achieves a remarkable improvement in the end-to-end delay. These protocols assume that all nodes are honest and cooperative.

In the area of information security, cryptographic primitives are often used to ensure properties such as confidentiality, integrity and so on. Several secure routing protocols [4] with cryptography have been proposed to protect *ad hoc* networks, such as SAODV [2] and Ariadne [3], but most of these protocols need centralised units or trusted third-parties to issue digital certificates or monitor network traffic. The common trusted authority actually violates the nature of self-organisation. Therefore these protocols are less practical for MANETs.

Recently, a new class of routing protocols in MANETs have been proposed, called trusted routing protocols, which consist of two parts: a routing strategy and a trust model [5]. The selection of next hops or forward paths in a routing strategy is made according to the trust model. Trusted-DSR [1] extended from DSR [24] selects a forward path based on a local evaluation of the trust values of all intermediate nodes along path to the destination. The node trust is calculated through an acknowledged mechanism from destination to source. Every acknowledged packet will increase the sender node's trusts in all the intermediate nodes along the path to the destination, whereas every retransmission decreases the trusts. It is impossible for senders to know which nodes discard packets. Pirzada *et al.* [26] evaluated the performance of three trust-based reactive routing protocols (trusted AODV, DSR and TORA) by varying the number of malicious nodes and other experiment settings. The results indicate that each trust-based routing protocol has its own advantage. Specifically, trust-based AODV routing maintains a stable throughput and surpasses TORA and DSR at higher traffic loads [26]. Therefore we combine our trust model with AODV to design a trust-based multipath routing protocol.

Inspired by the literature [7], Griffiths *et al.* [5] proposed the simple trusted AODV protocol (ST-AODV) in which the next-hop node whose trust value is greater than a constant threshold is selected as the forwarding node. The main differences between ST-AODV and our AOTDV are as follows: (i) ST-AODV is a single path routing protocol whereas AOTDV is a multipath protocol; and (ii) ST-AODV uses node trust values of next-hops to make routing choices, whereas AOTDV considers the trust values of paths as well as the number of hops. Therefore in AOTDV, the selected next hop indicates the shortest trusted path.

3 Trust model

Trust model essentially performs trust derivation, computation and application [26]. In our model, each node derives trusts in neighbours from packet forwarding ratio. During trust computation, a linear aggregation is used to estimate the overall trust in a node, and a continued product is used to compute the trust of a path. Trust applications including trust-based route discovery and route selection will be discussed in the next section.

3.1 Trust derivation

No matter what kind of trust models are used, two types of evolutions (direct trust and indirect trust) are available. Direct trust is first-hand information of neighbours and easy to obtain. Indirect trust is second-hand information about nodes, such as recommend trust from a third party. The indirect trust may incur additional communication cost for trust exchange. In order to simplify trust model, we only use the history of direct interactions among nodes to compute trust. In our trust model, passive acknowledgement is used as the single observable factor for assessing trust. Passive acknowledgement uses promiscuous mode to monitor neighbours' behaviours in the wireless radio channel, which allows a node to detect any transmitted packet in its communication range, irrelevant of their destinations.

We assume that after one node broadcasts a packet all neighbours will receive the packet correctly. However, if the distance between source and destination is beyond one hop, packets might be dropped by intermediate nodes because of unexpected causes (such as heavy traffic) or malicious attacks (such as black-hold or grey-hold attacks). Trust evaluation in a routing procedure is an assessment of forwarding behaviours of neighbours by a sender. More specifically, a node j will give its neighbour k a trust score after the node k transmits a packet sent by node j. Thus, we use packet forwarding ratio to evaluate the quality of forwarding.

Definition 1 (Forwarding ratio): Forwarding ratio is the proportion of the number of packets forwarded correctly to the number of those supposed to be forwarded. Correct forwarding means a forwarding node not only transmits a packet to its next hop node but also forwards devotedly (correct modification if required). For instance, when a malicious neighbour node forwards a data packet after tampering with data, it is not considered as correct forwarding. If the sender monitors this illegal modification, the forwarding ratio of the neighbour will decrease.

Definition 2 (Window forwarding ratio): The window forwarding ratio FR(t) is the packet forwarding ratio in a recent window. FR(t) is computed as follows

$$\mathrm{FR}(t) = \begin{cases} \dfrac{N_C(t) - N_C(t-W)}{N_A(t) - N_A(t-W)}, & t > W \\ \dfrac{N_C(t)}{N_A(t)}, & t \le W \end{cases} \tag{1}$$

where $N_C(t)$ represents the cumulative count of correct forwarding and $N_A(t)$ signifies the total count of all requesting before time t. The count of correct forwarding in a time window (from time $t - W$ to t) is equal to $N_C(t) - N_C(t - W)$, where W represents the width of the time window. We compute FR(t) only using the forwarding count and requesting count in the recent W time units. The history records out of the recent window are discarded.

In MANETs, all packets can be classified into two groups: control packets and data packets. Control packets are used for route request (discussed in Section 4.4.1), route reply (discussed in Section 4.4.2), route update (discussed in Section 4.5.1) and route error (discussed in Section 4.5.2). The accuracy of control packets plays a vital role in establishment of accurate routes in the network. So FR(t) is divided into two parts: control packet forwarding ratio, denoted by CFR(t), and data packet forwarding ratio, denoted by DFR(t). They are computed using forwarding count of control packets and data packets according to formula (1) respectively.

3.2 Computation of node trust

The trust of a node j in another node k (node trust for short) is a measure to ensure that packets sent by node j have actually been forwarded by node k. Two trust factors [CFR(t) and DFR(t)] are assigned weights in order to determine the overall trust value of a node. The direct trust in node k by node j is represented as T_{jk} and is given by the following formula

$$T_{jk}(t) = w_1 \times \mathrm{CFR}_{jk}(t) + w_2 \times \mathrm{DFR}_{jk}(t) \tag{2}$$

where $\mathrm{CFR}_{jk}(t)$ and $\mathrm{DFR}_{jk}(t)$ represent control packet forwarding ratio and data packet forwarding ratio observed by node j for forwarding node k at time t, respectively. The weights w_1 and w_2 (w_1, $w_2 \ge 0$ and $w_1 + w_2 = 1$) are assigned to CFR and DFR, respectively.

After each interaction, node j checks whether the neighbour k forwards the packet correctly. If so, the trust value T_{jk} increases. Otherwise, T_{jk} decreases. In our trust model, trust values are limited in a continuous range from 0 to 1 (i.e. $0 \le T_{jk} \le 1$). The trust value of 0 signifies complete distrust whereas the value of 1 implies absolute trust. An example of trust levels of nodes are listed in Table 1. If there is no interaction between two nodes, the initial trust value is set to 0.75 (less trustworthy node). That is, we adopt a limited optimistic view on unknown nodes. A threshold η, termed as the black-list trust threshold, is used to detect malicious nodes. In other words, if the trust value of a node is smaller than η, it will be regarded as a malicious node.

Each node, based upon its personal experiences, rewards collaborative nodes for their benevolent behaviours and punishes malicious nodes for their malevolent actions. To minimise the risk of transmission failure, nodes should interact with the trusted ones whose trust value is above the trust requirements of packets [37].

The sender places itself in promiscuous mode [27] after the transmission of any packet so as to overhear the retransmission by the forwarding node. Using this method, a node can know whether the packet that has been sent to a neighbour is indeed forwarded or not. The direct trust

Table 1 Trust levels of nodes

Level	Trust value	Meaning
1	$[0, \eta)$	malicious node
2	$[\eta, 0.75)$	suspect node
3	[0.75, 0.9)	less trustworthy node
4	[0.9, 1]	trustworthy node

values can also be shared among neighbours using a higher layer, such as reputation exchange protocol [28].

3.3 Computation of path trust

When a source discovers a path to the destination with the help of forwarding nodes, the trust value of the path (path trust) should be computed according to the trust values of nodes along the path. Considering the axiom [29] that concatenation propagation of trust does not increase trust, path trust should not be more than the trust values of intermediate nodes. So, at time t, the trust of a path P denoted by T_P (t) is equal to the continued product of node trust values in the path, that is

$$T_P(t) = \Pi(\{T_{jk}(t) | n_j, n_k \in P \quad \text{and} \quad n_j \rightarrow n_k \quad \text{and} \quad n_k \neq N_d\}) \tag{3}$$

in which, n_j and n_k are any two adjacent nodes among the path P, $n_j \rightarrow n_k$ means that n_k is the next-hop node of n_j and N_d is the destination node in the path P.

In particular, the node trust in the destination N_d by the next node (say node r) to the destination node is not required in formula (3). The trust T_{rd} denotes the weighted ratio of N_d forwarding packets from node r. The destination N_d should not forward the packets for itself, so T_{rd} is not used to compute the path trust to node d. If the destination node d is a neighbour of the source node s, the path trust is equal to 1 because we assume that all packets transmitted in one hop will reach the expected neighbour (node d).

As shown in Fig. 1*a*, the directed edge from node A to node B denotes a node trust ($T_{AB} = 0.8$). The trust value of path $P(A, B)$ is equal to 1 ($T_{P(A,B)} = 1$). The trust value of path $P(A, B, C, D)$ are equal to 0.72 (i.e. $T_{P(A,B,C,D)} = T_{AB} \times T_{BC} = 0.8 \times 0.9 = 0.72$). Fig. 1*b* shows a complex graph with multiple branches. There are three paths from A to F, in which the path $P(A, C, E, F)$ is the most trustworthy path.

The computation of path trust based on a continued product takes into account trust values of all intermediate nodes. Path trust denotes a joint probability at which packets will be forwarded if they are sent along the path. The computation of path trust complies with the opinion in information theory: the information cannot be increased via propagation [29].

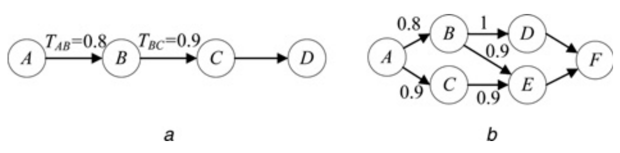


Figure 1 *Path trust computation*
a An example of a single path
b An example of a multiple path

4 Trust-based on-demand routing protocol

In this section, we describe an on-demand routing mechanism for *ad hoc* networks based on the proposed trust model. First, the structures of routing tables and trust record lists are depicted. Then, the procedures of route discovery and maintenance are discussed. Finally a sequence number method is given to avoid the routing loop.

4.1 Routing table

A routing table is used to store the routes to other nodes in an *ad hoc* network. Each node maintains a routing table composed of multiple routing entries. AOTDV adopts a hop-by-hop routing mechanism, in which the source is not expected to know all nodes in the path to a destination; it is sufficient for the source to know which neighbour is the next hop. When a data packet is going to a destination, it refers to local routing table to find the next hop (say node *j*). Once it reaches node *j*, it refers to node *j*'s routing table for the next hop to the destination. This process will continue until it reaches the destination.

Any node only stores routes to nodes that have interacted with it recently, not all routes in history because network topology of a MANET changes dynamically. The mobile nodes may join or quit the network at any time. An unused route to a destination in a certain period is considered invalid and will be deleted from the routing table.

Fig. 2 shows the structure of routing table entries for AODV [21], AOMDV [25] and AOTDV. The main difference between AOMDV and AOTDV is that path trust is added to the route list. A routing entry in AOTDV consists of the following fields:

1. *Destination:* the identifier of destination node.

2. *Destination sequence number:* the greatest known sequence number for destination denotes freshness of the route. It is used to avoid routing loop (discussed in Section 4.6).

3. *Next hop:* a neighbour node, to which a packet is sent.

4. *Hop count and path trust:* the two metrics compose an evaluation vector of a path. Different from single cost factor (e.g. AODV and AOMDV only use hop count as the cost), here the cost is hop count and trust constraint. The selected path is the shortest one in the paths which satisfy the packet trust requirement.

5. *Expiration timeout (ET):* the time after which the route is considered to be invalid. Each time a route is used to transmit

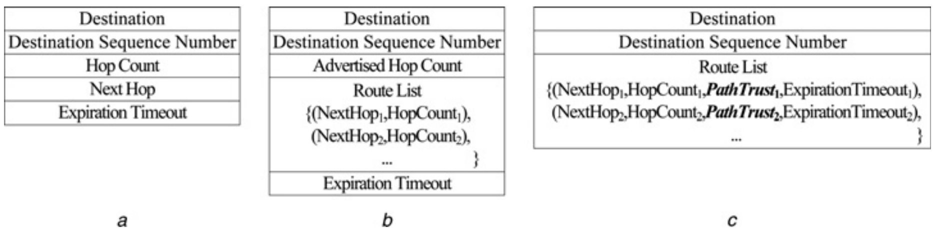

Destination
Destination Sequence Number
Hop Count
Next Hop
Expiration Timeout
a
Destination
Destination Sequence Number
Advertised Hop Count
Route List
{(NextHop$_1$,HopCount$_1$),
(NextHop$_2$,HopCount$_2$),
... }
Expiration Timeout
b
Destination
Destination Sequence Number
Route List
{(NextHop$_1$,HopCount$_1$,***PathTrust$_1$***,ExpirationTimeout$_1$),
(NextHop$_2$,HopCount$_2$,***PathTrust$_2$***,ExpirationTimeout$_2$),
... }
c

Figure 2 *Structure of routing table entries for AODV, AOMDV and AOTDV*

a AODV
b AOMDV
c AOTDV

data, the timeout for the route is reset to the current time plus a constant (active route timeout).

Multiple routes leading to the same destination are arranged in ascending order of HopCount, that is $\text{HopCount}_1 \leq \text{HopCount}_2 \leq \cdots \leq \text{HopCount}_n$. If two paths have the same HopCount, the one with greater PathTrust precedes, that is $\forall\ \text{HopCount}_i = \text{HopCount}_{i+1}$, $\text{PathTrust}_i \geq \text{PathTrust}_{i+1}$.

Table 2 gives an example of routing tables, in which there are two paths to node 5 and their next-hops are 1 and 2, respectively.

4.2 Trust record list

Trust record list is introduced to remember trust information. Each node will maintain a trust record for every neighbour to which packets have been sent for forwarding. A trust record listed in Table 3 contains a node ID, node trust, two integer counters of N_C and N_A for control packets, two integer counters of N_C and N_A for data packets and a packet buffer. The packet buffer is used to record all packets sent recently. It is a circular buffer, which means that the buffer will cycle and overwrite the oldest packet if it is not removed in time.

Table 2 Example of routing tables

Destination	Destination sequence #	Route list			
		Next hop	Hop count	Path trust	ET
1	1	1	1	1	5
5	2	1	3	0.6	4
		2	4	0.8	4

Table 3 Structure of a trust record

node ID
node Trust
N_C and N_A for control packets
N_C and N_A for data packets
packet buffer

Before sending a packet, a sender increases N_A for control packets or N_A for data packets by 1. For a broadcast packet including a route request packet or a route error packet, the sender increases N_A for control packets of all records in its trust record list except N_A for control packets of the node where the packet comes from. For a unicast packet, the sender only adds 1 to N_A for control packets of the next-hop when the packet is a route reply packet, or adds 1 to N_A for data packets of the next-hop when the packet is a data packet. To detect whether a packet is successfully forwarded, the sender will not delete the packet immediately after it is being sent. The packet will be stored in the packet buffer and wait for acknowledgement. A retry counter RetryCnt is used to remember retransmission number of every packet. If the sender in the promiscuous mode monitors that the packet is forwarded correctly, it will be removed from the packet buffer and the corresponding counter of correct forwarding (N_C) is increased by 1. The sender can update the node trust in the neighbour according to formulae (1) and (2). If the difference between the new value of node trust and the last value is greater than or equal to a threshold ζ, the node will broadcast a route update packet to its neighbours. The procedure will be discussed in Section 4.5.1.

4.3 Routing strategy

As shown in Fig. 3, the procedure of AOTDV routing is given as follows:

Step 1: Before a source s send a data packet to another node, say node d, the source looks up in the local routing table a route entry to node d. The qualified route should meet the trust requirement of the data packet. In other words, PathTrust of the qualified route is greater than the requirement of the data packet. If such routes are found, go to step 3.

Step 2: If there is not such a route, node s initiates a route discovery for d. If one or more paths are discovered, a route

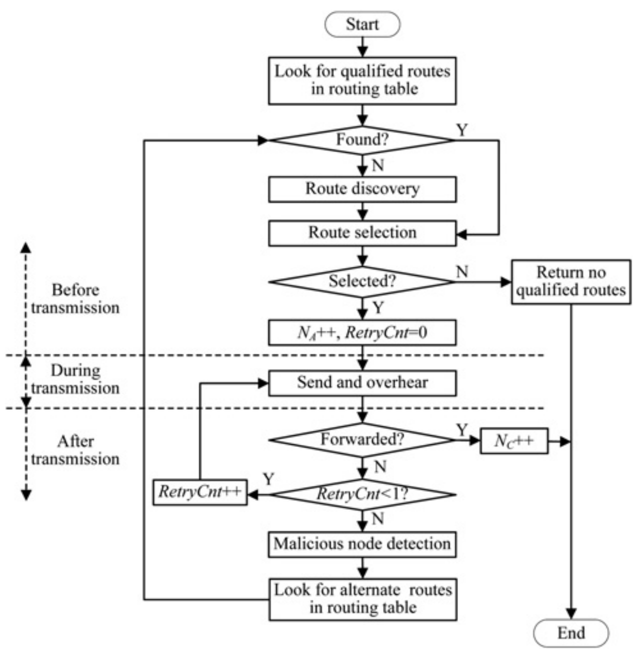


Figure 3 *Flowchart of routing procedure*

entry for these paths will be created and inserted into the routing table of node s.

Step 3: Node s selects the route with the smallest hop count in the qualified routes.

Step 4: If not a qualified route is selected, node s will return no qualified routes.

Step 5: If a qualified route is selected (assume that the selected next hop is node n), node s increases N_A for node n by 1, inserts the data packet into its trust record list, sets its retry counter to 0 and sends the packet. And then node s listens to the radio channel and checks whether the packet will be forwarded correctly by node n.

Step 6: If the packet is forwarded correctly by node n, node s increases N_C for node n by 1 and removes the packet in its trust record list. The procedure is over.

Step 7: If the packet is not forwarded correctly and its retry counter is less than 1, node s will increase the counter by 1 and retransmit the packet to node n. Go to step 6.

Step 8: If the counter is greater than or equal to 1, node s will detect malicious nodes according to its local trust record list and look for other valid routes in the routing table. Go to step 2.

In malicious nodes detection, if the node trust of a neighbour is smaller than the black-list trust threshold η, the neighbour will be regarded as a malicious node, and then be moved into a black list.

In particular, every node maintains a local black list. A malicious node in a black list is excluded by its neighbour holding the black list. That is, the packets from a malicious node will not be forwarded by the neighbour; meanwhile, the neighbour will not send packets to the malicious node except broadcast packets. If a node is evaluated very low by all its neighbours, any reply it gives to route requests is discarded, and any request it initiates is ignored. In other words, when a node exists in the black lists of all its neighbours, it will be excluded from the local network. If the node moves to a new sub-network, it will be regarded as a new comer with the initial trust value.

AOTDV is an on-demand multipath protocol, which tries to find multiple paths as candidates in one route discovery to reduce route discovery attempts. Only when all path candidates break or dissatisfy the trust requirement of a data packet, will one node try to discover a new route.

4.4 Route discovery and path selection

A route discovery is launched when no trusted routes exist. The node will initiate a network-wide flood by broadcasting a route request packet and wait for route reply packets. Every node maintains two monotonically increasing counters: a sequence number and a broadcast ID. Sequence number is used to supersede stale cached routes [21]. Broadcast ID is incremented whenever the source issues a new RREQ, RUPD or RERR packet.

4.4.1 Route request: An RREQ packet contains the following fields: ⟨Broadcastid, SourceAddr, Source SequenceNo, DestAddr, DestSequenceNo, HopCounter, RequiredTrust, ActualTrust⟩.

The first six fields are similar to the corresponding ones in AODV [21]. If the source does not have a route to the destination, DestSequenceNo is set to 0. If the source has a route, but its trust value is smaller than the trust requirement, the source will set DestSequenceNo of the RREQ to DestSequenceNo in its routing table. We add two fields – Required Trust (RT) and Actual Trust (AT). We use RT to represent the path trust value required by the data packet, which is set by the source and remains unchanged during the flood. The field AT denotes the continued product of trust values of nodes that the RREQ has passed in the route discovery. It is initialised to 1 by the source and varies with the transmission of the packet.

When an intermediate node j receives an RREQ from a neighbour k,

Step 1: It creates a route entry to node k with PathTrust as 1 if there is not a route to node k in its local routing table.

Step 2: It checks whether one copy of the same RREQ has been received. If so and the later copy has both greater HopCounter and less path trust (that is, the later path is farther and less trustworthy than the existing paths), the RREQ will be discarded and the procedure ends; otherwise, go to step 3.

Step 3: If node k is not the source, node j creates a reverse route to the source using the previous hop (node k) of the RREQ as the next hop. The path trust of the reverse route entry is set to $AT \times T_{jk}$ when T_{jk} is known, or min(AT, 0.75) when T_{jk} is unknown.

Step 4: If a valid route to the destination is available and SequenceNumber of the route is greater than the DestSequenceNo in the RREQ, node j generates an RREP to node k.

Step 5: Otherwise, node j modifies the AT of the RREQ using AT $\times$ T_{jk} when T_{jk} is known, or min(AT, 0.75) when T_{jk} is unknown. Then node j increases HopCounter by one and propagates the RREQ to all its neighbours.

If an intermediate node has a route entry for the desired destination, it determines whether the route is fresh by comparing the destination sequence number in its own route entry with the one in the RREQ. If the RREQ's sequence number for the destination is greater than or equal to that in the route entry, the intermediate node should not use its recorded routes to respond to the RREQ. Instead, it rebroadcasts the RREQ.

When the destination d receives an RREQ, it will compare the DestSequenceNo in the RREQ with the sequence number, say SN_d, maintained in node d. If DestSequenceNo is equal to SN_d, the destination will increase SN_d by 1. If DestSequenceNo is smaller than SN_d, the destination preserves SN_d. And then the destination decides whether to send back a route reply packet according to the condition discussed in the next subsection.

4.4.2 Route reply:

The intermediate node replies to an RREQ only when it has a route with a sequence number that is greater than that contained in the RREQ. If it does have a fresh route to the destination and the RREQ has not been processed previously, the node unicasts a route reply (RREP) packet back to its neighbour from which it received the RREQ. An RREP packet contains the following information:

⟨SourceAddr, SourceSequenceNo, DestAddr, DestSequenceNo, HopCounter, LifeTime, RequiredTrust, ActualTrust⟩.

The first six fields are similar to the corresponding ones in AODV [21]. The field RequiredTrust (RT) is equal to the RT in the RREQ. The field ActualTrust (AT) in an RREP denotes the continued product of trust values of nodes that the RREP passed during the process of route reply. And it is initialised to 1 by the destination.

If an intermediate node has multiple paths to the destination, it will reply two copies of RREP at most, one of which has the smallest hop count and the other has the greatest trust value.

If the destination receives multiple copies of RREQ, it will reply the first k paths at most, whose path trust values are greater than or equal to the RequiredTrust of the RREQ and which come from different neighbours of the destination. Here, we assume that an earlier-arrived packet comes along a shorter path. If several path trust values are smaller than the RequiredTrust, the destination will reply at most k of the shortest paths from different neighbours. The RREP contains the latest sequence number of the destination. The parameter k is used to control the number of RREPs and prevent an RREP storm. Nasipuri *et al.* [30] concluded that additional routes beyond a few provide only marginal benefit. So we assign 3 to k in our experiments.

If an intermediate node receives an RREP, it will unicast the RREP through the route whose PathTrust is not less than the RequiredTrust of the RREP and whose HopCount is the minimal in all paths to the source. As the RREP travels back to the source, each node along the path sets up a forwarding route to the destination and refills its ET for routes to the source and the destination.

4.4.3 A routing example:

In *ad hoc* networks on battlefields or business applications, different data have different requirements for importance and trust levels [37]. Generally, the more important data are, the more secure and trusted routes are needed. An example of three trust levels is listed in Table 4. At the beginning of route discovery, the trust value required by a packet is assigned to RequiredTrust of the request packet.

Fig. 4*a* shows the network topology and node trust values at a moment. The assumptions in the example are listed as follows:

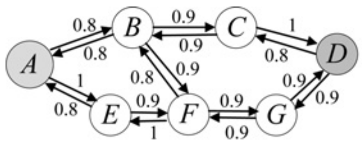
1. All nodes do not have any route to node D while node A tries to send a data packet to node D.

2. The packets require the trust of a path more than 0.7.

3. The sequence numbers of all nodes are equal to 5 and the BroadcastID of node A is equal to 8.

Table 4 Example of the trust requirements of data packets

Level	Trust value	Description
1	[0.6, 0.75)	less important data
2	[0.75, 0.9)	important data
3	[0.9, 1]	extremely important data



a

Dst	SN	RouteList			
		NH	HC	PT	ET
A	5	*A*	1	1	7
F	5	*F*	1	1	9

b

Legend:
Dst: Destination
SN: Dst. Sequence Number
NH: NextHop
HC: HopCount
PT: PathTrust
ET: Expiration Timeout

Dst	SN	RouteList			
		NH	HC	PT	ET
C	5	*C*	1	1	8
G	5	*G*	1	1	9
A	5	*C*	3	0.72	8
	5	*G*	4	0.81	9

c

Dst	SN	RouteList			
		NH	HC	PT	ET
A	5	*A*	1	1	5
F	5	*F*	1	1	7
C	5	*C*	1	1	9
D	5	*C*	2	0.9	9

d

Dst	SN	RouteList			
		NH	HC	PT	ET
B	5	*B*	1	1	7
E	5	*E*	1	1	9
D	5	*B*	3	0.72	7
	5	*E*	4	0.81	9

e

Figure 4 *Example of a route discovery*

a Node trust values
b Routing table of node *B* after receiving the latter RREQ
c Routing table of node *D* after receiving RREQ
d Routing table of node *B* after receiving RREP
e Routing table node *A* after receiving RREP

4. The maximum expiration time of a route entry is equal to 9 and each transmission of a packet costs one unit of time.

No matter which one of AODV [21], AOMDV [25] or AOTDV is adopted, node *A* will initiate a route discovery for node *D* and broadcast an RREQ to its neighbours.

Consider the procedure of the route discovery in AODV and AOMDV. The RREQ packet in AODV includes ⟨BroadcastID=8, SourceAddr = *A*, SourceSequenceNo = 5, DestAddr = *D*, DestSequence No = 0, HopCounter = 0⟩. The RREQ in AOMDV carries an additional field, called FirstHop, to indicate the first hop (a neighbour of the source) taken by the RREQ. The FirstHop of the RREQ along the path *P*(*A*, *B*, *C*, *D*) will be set to *B*. If there are multiple paths between an intermediate node and the node *A*, the intermediate node will receive multiple copies of the RREQ. Only the first arrived RREQ copy is forwarded in AODV and AOMDV. In the example, node *B* only forwards the RREQ from node *A*. Node *F* only forwards the first arrived copy of the RREQ from node *B* or node *E*. Assume that the copy from node *E* arrives at node *F* earlier. When intermediate nodes receive the RREQ, they will create a reverse route to the last-hop node and the source node *A*. One difference between AODV and AOMDV is that node *B* only creates one reverse path *P*(*B*, *A*) to node *A* in AODV whereas it creates two reverse paths *P*(*B*, *A*) and *P*(*B*, *F*, *E*, *A*) in AOMDV. Node *F* will also create two reverse paths to node *A* in AOMDV. Each time the RREQ passes an intermediate node, its HopCouter is increased by one. Two copies of the RREQ will arrive at the destination node *D* if no malicious nodes exist in the network. The destination *D* will receive two copies of the RREQ in AODV or AOMDV, in which one copy goes along the path *P*(*A*, *B*, *C*, *D*) and the other passes the path *P*(*A*, *E*, *F*, *G*, *D*). We assume that the copy along the former path arrives earlier because the hop count of the former (3) is less than the one of the latter (4). Node *D* only replies the earlier copy in AODV, whereas it replies the two copies in AOMDV because the two paths are link-disjoint [25]. At last, in AODV, the source node *A* receives one RREP from node *B* and creates a route to node *D* along the path *P*(*A*, *B*, *C*, *D*), whereas in AOMDV it receives two RREP packets and creates two routes along the path *P*(*A*, *B*, *C*, *D*) and the path *P*(*A*, *E*, *F*, *G*, *D*).

Next, we describe the discovery procedure in AOTDV. Node *A* broadcasts the RREQ carrying two additional fields: RequiredTrust(0.7) and ActualTrust(1). After node *B* receives the RREQ from node *A*, it will not only create a reverse route to node *A* but also rebroadcast the RREQ to its neighbours. In the RREQ transmitted by node *B*, ActualTrust is equal to $T_{P(B,A)}$, that is ActualTrust = 1. Node *B* will also receive another copy of the RREQ from node *F*, but the copy will be discarded because the trust of the path *P*(*B*, *F*, *E*, *A*), $T_{P(B,\ F,\ G,\ A)} = T_{P(F,\ E,\ A)} \times T_{BF}$ $1 \times 0.9 = 0.9$ is smaller than that of the path *P*(*B*, *A*) $T_{P(B,\ A)} = 1$, while path *P*(*B*, *F*, *E*, *A*) is longer. After node *B* receives the latter copy of the RREQ, it will create a route to node *F* and update the ET of the existing routes. The ET of the route to node *A* is changed to 7 because two units of time have elapsed. The routing table after node *B* receiving the second copy of the RREQ is showed in Fig. 4*b*. The node *F* will successively receive the RREQ from node *B* and node *E*. There are two cases for node *F*. Case (1): if node *F* receives the RREQ from node *B* earlier, it will broadcast the RREQ. In the RREQ, HopCounter is changed to 2 and ActualTrust is updated to ActualTrust × T_{FB} (i.e. 1×0.8). After receiving the second copy of the RREQ from node *E*, node *F* will also broadcast it because the latter copy indicates a path with greater trust ($T_{P(F,E,A)} = 1$) compared with the former one ($T_{P(F,B,A)} = 0.8$). Case (2): if node *F* receives the RREQ from node *E* earlier, it will broadcast the RREQ to node G. However, the copy of the RREQ from node B will be discarded because the two paths have the same hop count and the latter $P(F, B, A)$ has smaller path trust value than the former $P(F, E, A)$. The two cases may happen in theory, but we assume that case (2) occurs in the next discussion for simplification.

One copy of the RREQ will be forwarded by node C whereas another copy will be forwarded by node G. After receiving the first copy from node C, the destination D creates a route to node C and a route to node A and unicasts an RREP to node C. After one unit of time, node D receives the second copy of the RREQ from node G and then creates a route to node G. The trust value of $P(D, G, F, E, A)$ is equal to 0.81 and it is greater than that of $P(D, C, B, A)$. Therefore node D will create another route to node A, the next hop of which is node G.

At this moment, the routing table of node D is shown in Fig. 4c. Then, node D will unicast an RREP to node G. After receiving the RREP from node C, node B appends a route to node C and a route to node D to its routing table. As shown in Fig. 4d, the route entry to node D is $\langle$NextHop=C, HopCount = 2, PathTrust = 0.9, ExpirationTimeout = 9$\rangle$. At last, node A receives two copies of the RREP from nodes B and E, and creates two routes to node D. After node A receives the second copy, its routing table is shown in Fig. 4e, in which one route entry (named r_1) is $\langle$NextHop=B, HopCount = 3, PathTrust = 0.72, ExpirationTimeout = 7$\rangle$, and the other (named r_2) is $\langle$NextHop=E, HopCount = 4, PathTrust = 0.81, ExpirationTimeout = 9$\rangle$.

After the route discovery, node A finds two trusted paths to node D and their trusts are greater than the requirement (0.7). Node A will choose node B rather than node E as the next hop because the route r_1 has shorter distance than the route r_2. Node B receives the packet from node A and looks a shortest path up in its routing table to node D. Node B will select node C as the next hop to forward the packet. Finally, the packet comes along the path ($A \rightarrow B \rightarrow C \rightarrow D$) to its destination. If another data packet requires a path to node D with path trust value not less than 0.8, the path ($A \rightarrow E \rightarrow F \rightarrow G \rightarrow D$) will be selected.

4.5 Route maintenance

The route maintenance in AOTDV is similar to that in AODV. A node maintains and updates its routing table when receiving an RREQ, RREP or route error (RERR) packet. In addition, a new route update (RUPD) packet is used to update the hop count and path trust of a route.

4.5.1 Route update: An RUPD packet contains the following fields: <BroadcastId, SourceAddr, Source SequenceNo, UpdateDestAddr, UpdateDestSequenceNo, UpdateHopCounter, UpdatePathTrust>, where UpdateHop Counter represents the count of nodes in the path and UpdatePathTrust denotes the new trust value of the path from SourceAddr to UpdateDestAddr.

If the increase or decrease of trust in node k by node j is greater than or equal to the trust update threshold ζ, that is $\Delta T_{jk} \geq \zeta$. Node j will broadcast an RUPD packet to its neighbours. First, node j looks up in the routing table the set of routes X in which the next hop is node k and destination is not node k. Node j update all path trust values in the set X. Then, node j broadcasts an RUPD packet in which UpdateDestAddr is the destination ID passed by node k and UpdateDestSequenceNo is SequenceNumber of the destination in j's routing table. UpdatePathTrust in the RUPD is set to the new value of PathTrust in the route.

When a node i receives an RUPD packet, it will check whether its UpdateDestAddr is node i. If so, the RUPD packet will be discarded. Otherwise, node i checks whether any route to UpdateDestAddr exists in its routing table. If not, node i adds a new route entry to UpdateDestAddr and discards the RUPD. If a route to the UpdateDestAddr exists, node i refreshes its routing table and broadcasts an RUPD in which the source is node i and other fields are set according to the updated route entry.

For instance, we set the update threshold ζ to 0.1. If the node B in Fig. 4a detects that trust T_{BC} is changed from 0.9 to 0.8, it will look up in its routing table those routes in which the next hop is node C and the destination is not node C. In Fig. 4d, there is only one route meeting the two conditions. The destination of the route is node D. So, node B broadcasts an RUPD with UpdateDestAddr = D and UpdatePathTrust = 1 × 0.8 = 0.8. After node A receives the RUPD, it computes a new PathTrust of the route to node D as UpdatePathTrust(0.8) × T_{AB}(0.8) and changes PathTrust to 0.64.

Broadcasting route update packets in case of a trust value increase could help to reduce the count of route discovery. For example, in Fig. 4e, there are two routes to node D in the routing table of node A: route 1 ($A, B, \ldots, D$) and route 2 ($A, E, \ldots, D$). Assume the trust update threshold ζ, is set to 0.5. If T_{EF} is changed from 0.9 to 0.95, node E will broadcast a route update packet (UpdatePathTrust = 0.855) to node A. After receiving the route update packet, node A will change the path trust (PT) of route 2 with from 0.81 to 0.855(UpdatePathTrust × T_{AE} = 0.855 × 1). If there is a new packet from A to D which requires trust value above 0.85, the node A will choose node E as next hop. If there no route update packets in case of a trust value increase, node A has not known the change in path trust and has to launch a new route discovery.

4.5.2 Route error: When a link failure is detected (by a link layer feedback, for example), an RERR is sent back to all sources using that broken link.

An RERR packet contains the following fields: <BroadcastId, SourceAddr, SourceSequenceNo, DestCount, UnreachableDestList>, where DestCount represents the count of unreachable nodes and UnreachableDestList is composed of the address and the sequence number of unreachable nodes in the source node's neighbours.

The corresponding route entries are removed by an RERR along its way. Only when there is not a qualified route entry to the destination that a node needs to send packets to, the node initiates a new route discovery.

4.6 Loop freedom

All protocols using broadcast to discover routes will encounter routing loops. For example, an intermediate node j broadcasts an RREQ to its neighbours. A neighbour, say node k, receives the RREQ, creates a reverse path to the source via node j, and broadcasts it, which in turn is heard by node j. If node j accepts the RREQ and forms a reverse path to the source via node k, this will form a loop route. When more than one path exists from a source s to an internal node i, multiple copies of the RREQ packet will arrive at node i. Node i forwarding all such copies will lead to routing loops because node i could re-broadcasts the same copy of RREQ that has been forwarded before.

Sequence numbers in AODV play a key role in ensuring loop freedom [25]. Therefore in order to avoid any possibility of loops, every node maintains a monotonically increasing sequence number for itself. It also maintains the highest known sequence numbers for each destination in the routing table (called "SequenceNumber" in Section 4.1). Destination sequence numbers are tagged on all routing packets, thus providing a mechanism to compute the relative freshness of two copies of routing packets generated by two different nodes for the same destination.

Only when having received a fresh control packet of RREQ or RREP, a node will create a reverse path to the source or a forward path to the destination. A node should not create or update a route on the arrival of an old control packet. Assume that a node j receives a control packet to a destination d ($d \neq j$) from a neighbour k. Let the variables SeqNumber_k^d, HopCounter_k^d and ActualTrust_k^d represent the DestSequenceNo, HopCounter and ActualTrust of the control packet, respectively. Let SeqNumber_j^d, RouteList_j^d, MaxTrust_j^d and MinHops_j^d be SequenceNumber, RouteList, maximum PathTrust and minimum HopCount of multiple paths to destination d in the routing table of node j, respectively. The update rule for routing table in AOTDV is given as Fig. 5.

The route update rule above is invoked on receiving an RREQ or an RREP packet. Lines 1, 8 and 9 of the route update rule ensure loop freedom (see the Appendix). The protocol only allows accepting alternate routes with smaller hop count or ones with higher path trust.

5 Experimental results

To evaluate the performance of AOTDV, AODV [21] and AOMDV [25], we have conducted a comprehensive test using NS-2 network simulator [31]. All experiments are carried out on a PC machine with a Pentium 4 processor (2.4 GHz) and 2 GB main memory.

5.1 Experiment setup

NS-2 simulator (Version 2.34) is used to evaluate the performance of these on-demand routing protocols in different conditions. The distributed coordination function of IEEE 802.11 [32] for wireless LANs is adopted as the MAC layer protocol. We take an unslotted Carrier Sense Multiple Access protocol with Collision Avoidance [33] to transmit data packets as well routing packets. Within a rectangular field of 1000 $\times$ 1000 m, we dispersed 50 nodes randomly whose transmission radius of every node in one hop is fixed at 250 m. The node mobility uses the random waypoint model [34] in which each packet starts its journey from a location to another at a randomly chosen speed. A maximum speed of 0 m/s implies that the MANET is a static network. Once the destination is reached, another destination is randomly chosen after a pause time. The fixed simulation parameters in NS-2 are listed in Table 5.

Table 6 shows the simulation parameters in AOTDV which are set as follows:

1. Trust update threshold ζ is set between 0.02 and 0.1.

2. The black-list trust threshold η is set between 0.1 and 0.5.

3. The constant W (the width of window) in formula (1) is set to 300 s. Packet forwarding ratio is computed by the cumulative count of correctly forwarded packets and the number of all requests from the beginning. The weights for control packets and data packets would be set according to two settings: (1) $w_1 = 0.6$ and $w_2 = 0.4$; (2) $w_1 = 0.5$ and $w_2 = 0.5$. In setting (1), the forwarding ratio of control packets is considered more important than that of data packets. In setting (2), they have equal importance. Accordingly, AOTDV with setting (1) and (2) is represented as AOTDV-1 and AOTDV-2, respectively.

Malicious nodes can launch modification attacks, grey-hole attacks or black hole attacks. In the experiments, malicious nodes carry out modification attacks by altering source addresses in the control packets or data packets. In grey-hole attacks, malicious nodes selectively forward data packets at a ratio set to a constant of 30%. In black hole attacks, malicious nodes drop all data packets, but deliver route request and reply packets devotedly in order to participate in data transmission. The fractions of modification, grey hole and black hole attacks are 30, 40 and 30%, respectively.

```
1.  if ($SeqNumber_j^d < SeqNumber_k^d$) then
        //a fresher control packet
2.      $SeqNumber_j^d = SeqNumber_k^d$;
3.      $RouteList_j^d$ =NULL;
4.      if ($T_{jk}$ is unknown) then $NewActualTrust = min(ActualTrust_k^d, 0.75)$;
5.      else $NewActualTrust = ActualTrust_k^d \times T_{jk}$;
6.      endif
7.      insert ($k$, $HopCounter_k + 1$, $NewActualTrust$) into $RouteList_j^d$;
8.  elseif ($SeqNumber_j^d = SeqNumber_k^d$) then
9.      if ($ActualTrust_k^d \times T_{jk} > MaxTrust_j^d$) or ($HopCounter_k^d < MinHops_j^d$) then
10.         if ($T_{jk}$ is unknown) then $NewActualTrust = min(ActualTrust_k^d, 0.75)$;
11.         else $NewActualTrust = ActualTrust_k^d \times T_{jk}$;
12.         endif
13.         insert ($k$, $HopCounter_k + 1$, $NewActualTrust$) into $RouteList_j^d$;
14.     endif
15. endif
```

Figure 5 *AOTDV route update rule*

Table 5 Fixed simulation parameters

Parameter	Value
simulation time	500 s
number of nodes	50
map size	1000 × 1000 m
mobility model	random way point
traffic type	constant bit rate (CBR)/UDP
transmission radius	250 m
packet size	512 bytes
connection rate	4 pkts/s
connections	20
pause time	10 s

Table 6 Varying simulation parameters

Test no.	Malicious nodes	Max speed (m/s)	Trust update threshold	Black-list trust threshold
1	10	0–30	0.05	0.4
2	0–20	10	0.05	0.4
3	10	10	0.05	0.4
4	20	10	0.02–0.1	0.1–0.5

5.2 Performance metrics

We use five metrics to evaluate the performance of the routing protocols, in which the first two metrics are the most important for best effort route and transmit protocols.

1. *Packet delivery ratio:* or packet throughput, the fraction of the data packets delivered to destination nodes to those sent by source nodes.

2. *Average end-to-end latency:* the average time taken by the data packets from sources to destinations, including buffer delays during a route discovery, queuing delays at interface queues, retransmission delays at MAC layer and propagation time.

3. *Routing packet overhead:* the ratio of the number of control packets (including route request/reply/update/error packets) to the number of data packets.

4. *Path optimality:* the proportion of the total number of hops in the shortest paths to the one of hops in the paths taken by data packets.

5. *Detection ratio:* the ratio of the number of nodes whose behaviour (malicious or benevolent) is identified correctly to the actual number of such nodes in the network. This metrics is only used to evaluate the performance of AOTDV under different settings.

To decrease the disturbance of random error, every experiment repeats 50 times and the average experiment results are computed.

5.3 Test 1: varying node speeds

In the first test, we compare AOTDV with AODV and AOMDV as the maximum speed of nodes varies from 0 to 30 m/s. As shown in Fig. 6*a*, the delivery ratios of AODV and AOMDV decline remarkably as nodes speed up whereas the delivery ratios of AOTDV-1 and AOTDV-2 decrease gently. The differences become more apparent at higher speeds. This advancement of AOTDV can attribute to the improved probability of node behaviour detection because of more interactions. Under a lower speed (speed < 10 m/s), both AOTDV and AOMDV can make use of the multipath feature, which elevates the probability of successful delivery to a trustworthy node. In contrast, a node in AODV only maintains one route to a destination and is unable to improve packet delivery in case of route break. AOTDV-1 has higher delivery radios than AOTDV-2 because the former pays more attention to control packets and alleviates the impacts of malicious nodes in route discoveries.

Figs. 6*b* and *c* illustrate that the average end-to-end latency and routing packet overhead in these protocols rise with the

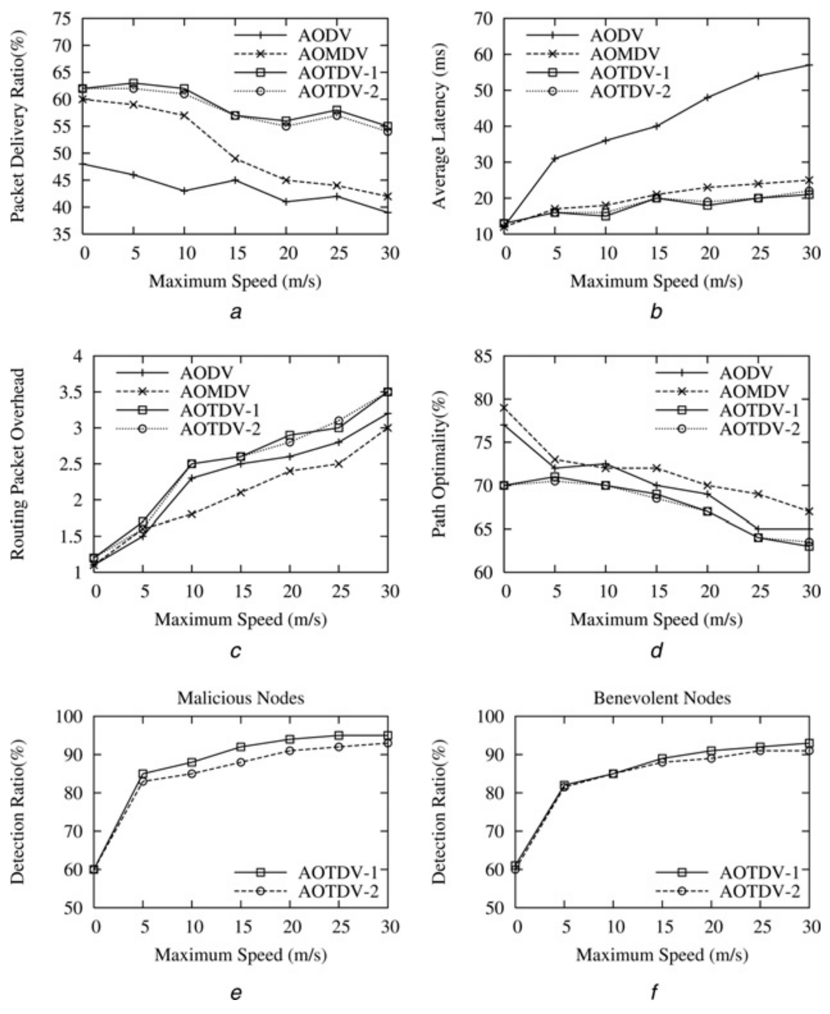


Figure 6 *Test 1: Performance with varying node maximum speeds*

a Packet delivery ratio
b Average end-to-end latency
c Routing packet overhead
d Path optimality
e Detection ratio of malicious nodes
f Detection ratio of benevolent nodes

increase of maximum speed. At higher speeds, route entries become invalid more quickly and thus source nodes initiate more route rediscoveries before sending data. At the highest speed of 30 m/s, the average latency and routing packet overhead reach their peaks, respectively. AOTDV has a little lower average latency (2–6 ms) than AOMDV when the speed is greater than 5 m/s. The routing overhead in AOTDV and AOMDV remains comparatively lower than that in AODV. Because multiple paths are found in one route discovery, the frequency of route discovery is smaller in AOTDV and AOMDV than in AODV. The overhead in AOTDV is higher than that in AODV and AOMDV. There are two reasons for the higher overhead: (i) more RREQ and RREP packets need to be sent for qualified routes to meet trust requirement in AOTDV, and meanwhile, trust requirement is not considered in AODV and AOMDV; and (ii) the additional route update packets increase the amount of control packets and the routing packet overhead in AOTDV.

As shown in Fig. 6*d*, the path optimality of these protocols degrades as the speed increases. AOTDV has smaller path optimality than AODV and AOMDV. It is observed that the actual paths may not be the best available paths because of the fact that in AOTDV intermediate nodes make routing selection considering hop count and trust value. Sometime only longer paths have satisfied trust demands. They reduce the path optimality and also increase a little latency of the packet transmission.

The detection ratio of AOTDV increases with node speed as shown in Figs. 6*e* and *f*. We can observe that when the nodes move faster, the number of interactions between nodes increases gradually. This leads to higher detection ratios of both malicious nodes and benevolent nodes. In general, AOTDV-1 is a little better than AOTDV-2 in terms of detection ratio. This attributes to the greater weight of control packets. Especially, at higher speed, AOTDV-1 has better detection ratios.

5.4 Test 2: varying number of malicious nodes

In test 2, we evaluate these protocols by varying number of malicious nodes. When there are no malicious nodes, the packet loss rate is about 1% in AODV, AOMDV and AOTDV. As shown in Fig. 7*a*, the delivery ratios in all protocols degrade sharply as the number of malicious nodes increases. The delivery ratio of AOTDV drops from 99 to 54% as the number of malicious nodes varies from 0 to 20. Lower packet delivery ratio means less network throughput. Malicious nodes essentially limit interactions between nodes in the network. However, in AOTDV, intermediate nodes have several routes to a destination so that when detecting grey hole or black hole attacks, they can try alternative routes to forward packets and thus packet delivery ratio is improved. The delivery radio of AOTDV-2 is very close to that of AOTDV-1, but the latter is a bit better.

As shown in Fig. 7*b*, the average latency of all three protocols declines slowly with the increase in number of malicious nodes. There is an obvious reduction in the average latency with AOMDV and AOTDV compared to AODV. This is because the availability of alternative routes eliminates delay caused by route rediscoveries. These multiple candidates contribute to reduce the end-to-end latency.

When the number of malicious nodes increases to 20 (40% of the whole nodes), the routing packet overhead of AOTDV is approximately 3.5 as shown in Fig. 7*c*. Normally, both AODV and AOTDV generate about 2.4 control packets on average for every data packet whereas AOMDV creates about 1.7 control packets. The increased control packets in AOTDV are primarily because of its route discovery mechanism that broadcasts more RREP packets to look for trustworthy routes to destinations.

As shown in Fig. 7*d*, AODV in these protocols exhibits the best path optimality (86%) when there are no malicious nodes. As malicious nodes increase, the path optimality of all three protocols decreases. On the whole, the path optimality of AOTDV is smaller than that of AOMDV. AOTDV is able to detect and filter out malicious nodes and find trustworthy paths to destinations even though these paths are not the shortest ones.

The detection ratios in both versions of AOTDV are shown in Figs. 7*e* and *f*. They decline with the increment of the number of malicious nodes. It is observed that the more malicious nodes are, the more serious their damage is. Accordingly, the detection is harder. For the both types of detection, a ratio of over 70% is maintained if the percentage of malicious nodes is not more than 40%. Overall, AOTDV-1 is better than AOTDV-2 in the detection ratios, especially when more malicious nodes exist in the network.

5.5 Test 3: varying trust update threshold

Trust update threshold, an important constant for route maintenance, denotes the condition when update packets will be sent. For example, when the trust update threshold is set to 0.02, if the increase or decrease of a node trust is greater than or equal to the threshold, an RUPD packet will be broadcasted. In the third test, we evaluate the performance of AOTDV under setting (1) and setting (2) as the trust update threshold ζ increases from 0.02 to 0.1.

As shown in Fig. 8*a*, the delivery ratios in both AOTDV-1 and AOTDV-2 degrade slowly from 65 to 57% as the trust update threshold increases. When the threshold is set to a smaller value, the nodes' response to trust update is more sensitive. Accordingly, unqualified routes will be detected and bypassed in time. A source node could timely look up a substitute route in its local routing table or find new

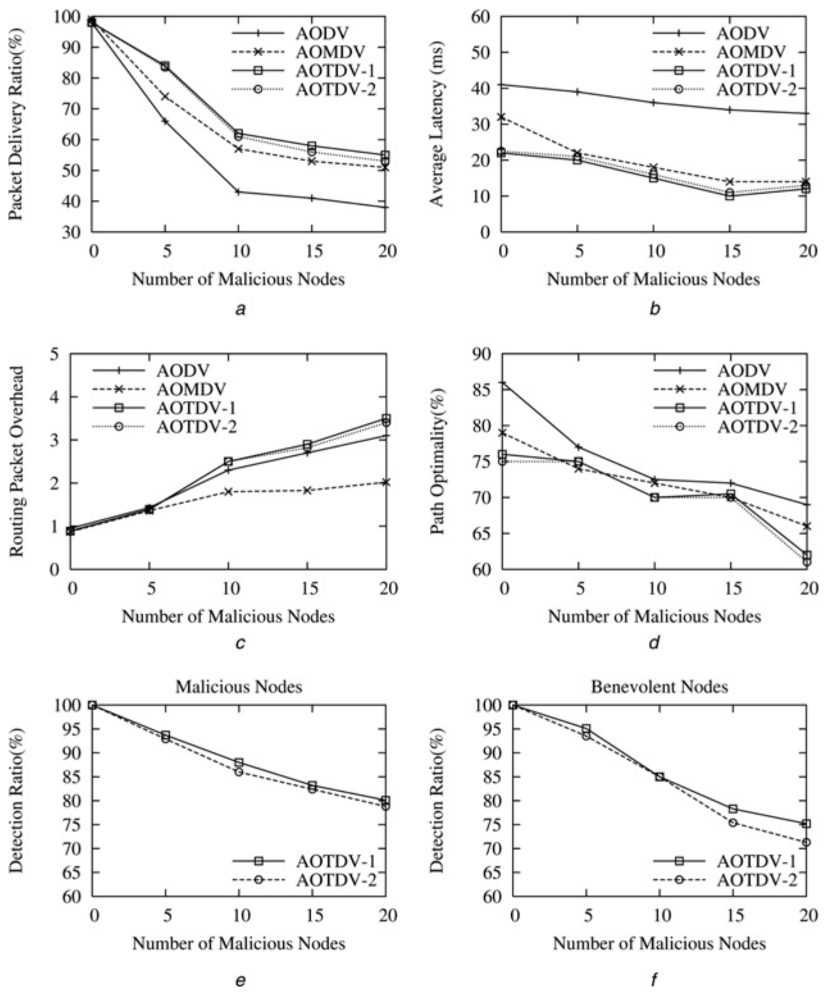


Figure 7 *Test 2: performance with a varying number of malicious nodes*

a Packet delivery ratio
b Average end-to-end latency
c Routing packet overhead
d Path optimality
e Detection ratio of malicious nodes
f Detection ratio of benevolent nodes

paths in a new route discovery. When the trust update threshold is set to a greater value, the update packets will decrease. Because nodes are not informed the trust changes smaller than the threshold, some packets would be sent along an unqualified route that had been trustworthy. These packets discarded by intermediate nodes increase as the trust update threshold goes up. However, as revealed in Fig. 8*b*, the average latency of AOTDV declines as the trust update threshold increases. The buffer delays and queue delays of all packets decrease when the number of update packets reduces. On the whole, AOTDV-1 has a little better performance than AOTDV-2.

When the threshold is set to a smaller value, more update packets will be transmitted. As shown in Fig. 8*c*, the percentage of control packets, that is routing packet overhead, declines as the trust update threshold increases from 0.02 to 0.1. When the threshold increases to 0.06, the routing packet overhead remains at about 2.5. Accordingly, the overhead of AOTDV-2 is less than AOTDV-1. Because AOTDV-2 is less sensitive to forwarding ratio of control packets than AOTDV-1, the frequency of path trust fluctuation is smaller in AOTDV-2.

As shown in Fig. 8*d*, the path optimality has changed a little as the trust update threshold rises to 0.1. The optimality ranges from 74 to 70%. We can observe that the trust update threshold has a strongly effect on changes of path trust, but it has little effect on hop counts of selected paths.

As shown in Fig. 8*e*, the increasing trust update threshold results in a reduction in the detection ratio for malicious nodes. The detection ratio for benevolent nodes in Fig. 8*f* also declines as the trust update threshold increases. The detection ratio of benevolent nodes is smaller than that of malicious nodes. This is because some normal nodes could be recognised to malicious nodes. For example, some nodes

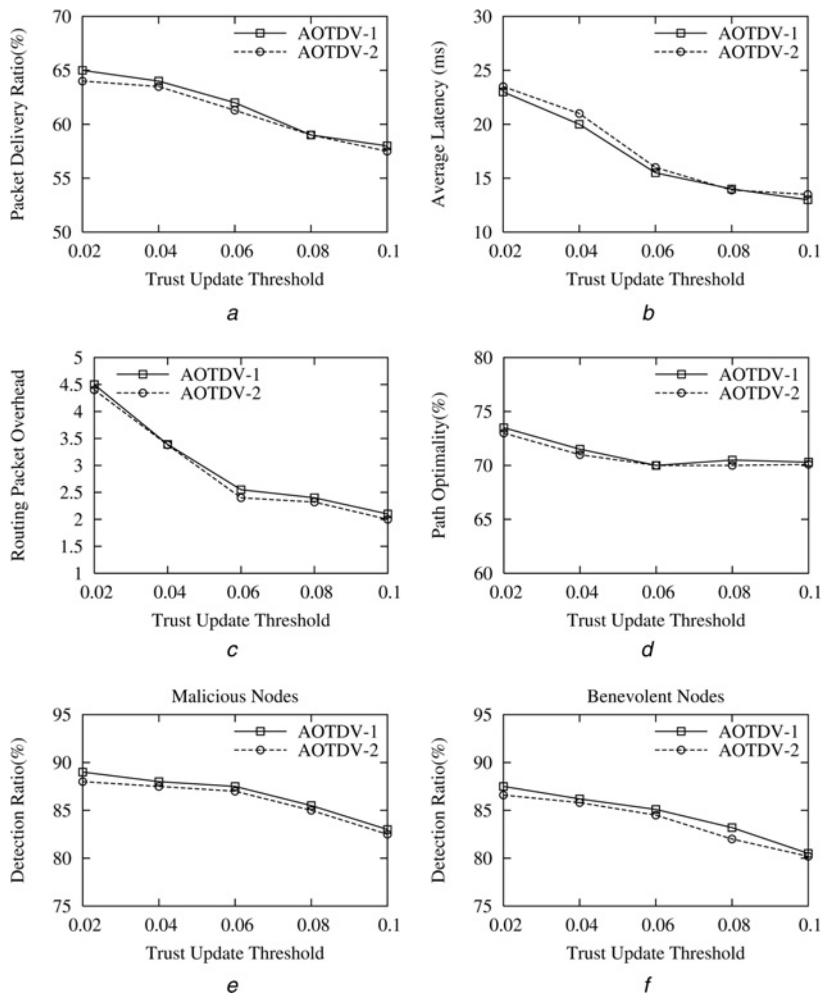


Figure 8 *Test 3: performance with varying trust update threshold*

a Packet delivery ratio
b Average end-to-end latency
c Routing packet overhead
d Path optimality
e Detection ratio of malicious nodes
f Detection ratio of benevolent nodes

move out of the range of the senders so that their forwarding ratios decrease. These nodes would lose neighbours' trust and even be regarded as malicious nodes. For the both types of detection, a ratio of over 80% is maintained when the trust update threshold ranges from 0.02 to 0.1. Regardless of the detections for malicious nodes or benevolent nodes, AOTDV-1 achieves a better precision than AOTDV-2.

5.6 *Test 4: varying black-list trust threshold*

In the last test, we compare the two AOTDV versions with different trust threshold for local black lists. The number of malicious nodes is set to 20 and the trust threshold ranges from 0.1 to 0.5.

As shown in Fig. 9*a*, on the whole, the delivery ratios of the two versions are not as high as expected. In fact, they are smaller than 55%. This is because the proportion of malicious nodes is 40% and a lot of packets are not forwarded devotedly by the intermediate nodes. The delivery ratios of AOTDV-1 and AOTDV-2 increase slowly from 50 to 55% as the black-list trust threshold increases from 0.1 to 0.5. A smaller trust threshold means that more packets could be dropped by a node before it is regarded as a malicious node. When the threshold is equal to 0.4, the delivery ratios of AOTDV-1 and AOTDV-2 reach their peaks, respectively (i.e. 54.8% for AOTDV-1 and 54.2% for AOTDV-2).

Fig. 9*b* shows the average end-to-end latency of AOTDV as black-list trust threshold varies. The results indicate that the average latency increases gradually from 10.2 to 15.5 ms as the trust threshold ranges from 0.1 to 0.5. As the trust threshold is set to a smaller value, fewer nodes will be added to black lists. This leads to lower average latency at the smaller threshold. The average latency of AOTDV-1 is very close to that of AOTDV-2.

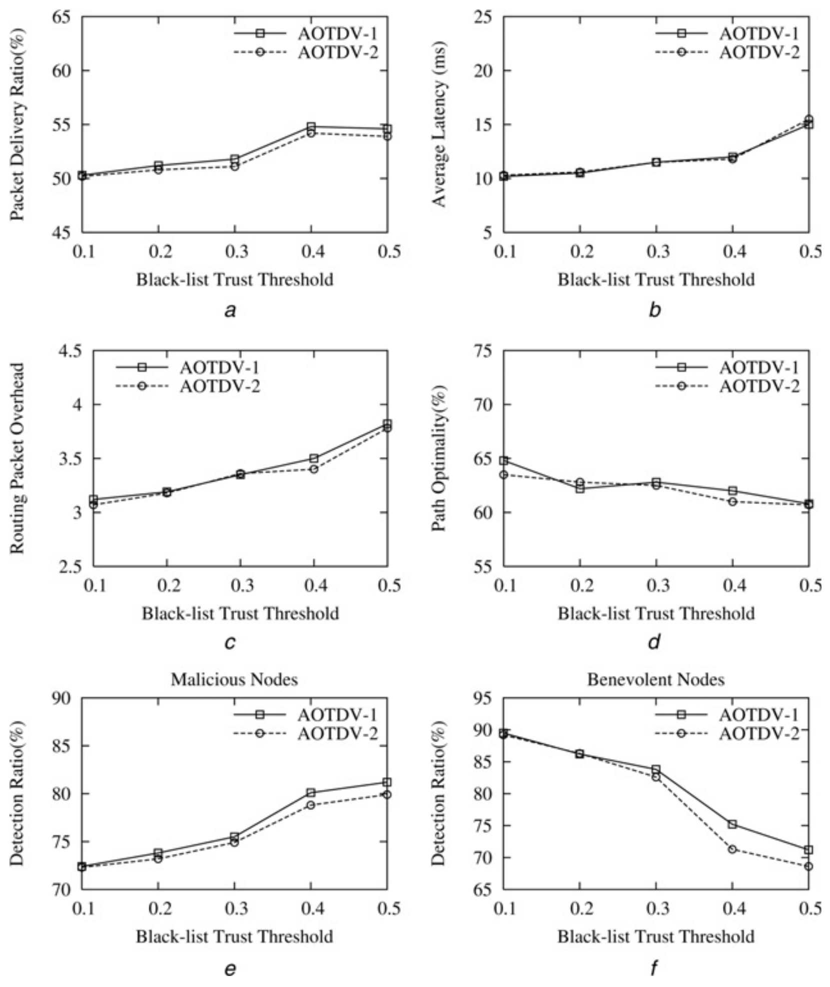


Figure 9 *Test 4: performance with varying black-list trust threshold*

a Packet delivery ratio
b Average end-to-end latency
c Routing packet overhead
d Path optimality
e Detection ratio of malicious nodes
f Detection ratio of benevolent nodes

The effects on the routing packet overhead and path optimality are shown in Figs. 9*c* and *d*, respectively. The average overhead is about 3.4 whereas the average path optimality is about 63%. If the trust threshold is assigned to a small value ($\eta < 0.3$), the malicious nodes launching grey hole attacks will not be detected and the count of route discoveries for avoiding malicious nodes is small. As the trust threshold increases, more nodes are detected as malicious nodes and more routing packets are forwarded along trustworthy paths. Accordingly, the routing packet overhead tends to rise with the increase in the trust threshold. The optimality of AOTDV-1 and AOTDV-2 fluctuates in the range from 62 to 65% as the trust threshold increases. Overall, AOTDV-1 and AOTDV-2 shows similar performances in routing packet overhead and path optimality.

Figs. 9*e* and *f* show the detection ratios for malicious nodes and benevolent nodes respectively. In the view of the trend, the changes of the two ratios are just opposite as the black-list trust threshold ranges from 0.1 to 0.5. The former ratio increases from 72 to 81% whereas the latter declines from 90 to 70%. The increase in the ratio for malicious nodes implies that when the trust threshold is set to a greater value, malicious nodes are easier to be detected. Unfortunately, more benevolent nodes would be considered as malicious nodes. Accordingly, the detection ratio for benevolent nodes reduces with the increment of thrust threshold. It should be pointed that the detection ratios change obviously when the threshold changes from 0.3 to 0.4. It is a coincidence that the delivery ratio, in Fig. 9*a*, also increases sharply (about 3%) during the period. This is because a constant 30% is used as the forwarding ratio of grey-hole nodes. When the black-list trust threshold is smaller than 0.3, the malicious nodes launching grey-hole attacks cannot be recognised. As the trust threshold rises to 0.4, AOTDV-1 shows better performance in detection ratios for malicious nodes about 3% and for benevolent nodes about 4% than AOTDV-2.

5.7 Summary

The experiment results in tests 1 and 2 show that AOTDV performs better than AODV and AOMDV as AOTDV gives higher delivery ratio and lower end-to-end latency. However, the benefit of AOTDV is obtained at the cost of the routing packets overhead. In other words, control packets increase greatly as the performance, especially, packet delivery ratio, improves in AOTDV. In tests 3 and 4, we evaluate the effects of the trust update threshold and the black-list trust threshold in AOTDV protocol. The analysis shows that the trust update threshold and black-list trust threshold should be maintained at an appropriate median (0.4–0.5) in order to obtain a trade-off among delivery ratio, latency, overhead and detection ratios. Overall, AOTDV-1 shows a better performance than AOTDV-2. That proves our observation that control packets play a more important role than data packets in MANETs.

6 Discussion

6.1 Trust derivation

Observing nodes' behaviours is an effective mechanism to determine whether a node can be trusted. Literatures [5, 7] use several aspects of node behaviours including acknowledgements, packet precision, gratuitous route replies and so on. These observations can be combined together to compute a trust value. Our opinion is that the behaviours of malicious nodes can also be detected using a much simpler scheme. In this paper, we build trust models using passive acknowledgement as the only observable factor for evaluating trust. We find that passive acknowledgement gives an effective indication of a node's behaviour of cooperation. The trust value of one node to another node summarises the former's experience with the latter's ability of delivering packets correctly. A node trust can be considered as a subjective measurement of a node's quality of forwarding, while a path trust can be used to anticipate the quality of forwarding packets along the path.

The node's trust can also derive from indirect trust – recommendation. Recommendation mechanism is an important component in any trust evaluation system [29]. The effectiveness of recommendation is closely related with communication overhead and mobility. Routing protocols should be simple and low communication cost, especially in mobile and resource-limited networks. So recommendation is considered in this paper.

6.2 Attacks on trust-based routing protocols

A sender places itself in a promiscuous mode after the transmission of any packet so as to overhear the retransmission by the forwarder. The passive acknowledgement provides AOTDV with the abilities against several attacks from malicious nodes, including black-hole attack, modification attack, fabrication attack and impersonation attack.

However, MANETs are vulnerable to the Sybil attack and newcomer attack. If a malicious node can create several faked IDs, the trust evaluation system might suffer from the Sybil attack [35, 36]. Faked nodes can share or even take the blame. One approach to prevent Sybil attack is to create a trusted agency to authenticate identities. Without the logically centralised authority, Sybil attack is always possible except under extreme and unrealistic assumptions of resource parity and coordination among entities [35]. In addition, if a malicious node can easily register as a new user, the trust evaluation suffers from the newcomer attack. Here, malicious nodes can easily remove their bad history by registering as a new user. The defense against the Sybil attack and newcomer attack does not rely on the design of trust evaluation system, but the authentication and access control mechanisms, which make registering a new ID or a faked ID difficult.

6.3 Black list

A node with trust value smaller than the black-list trust threshold will be regarded as a malicious node and added to a local black list. The malicious nodes in a black list are only excluded by the node holding the black list. If a node exists in the black lists of all its neighbours, it will be excluded from the local network. If the node moves to a new sub-network, it will be regarded as a new comer with initial trust value (less trustworthy node). This is a simple but strict rule.

If the behaviour of nodes in local black list can be re-evaluated, a routing protocol can select a 'good' node as main forwarding node and a malicious node in the black list as a redundant forwarding node. Multiple copies of data packets are transmitted in parallel at the same time. This will make the route protocol complicated.

Punishing malicious nodes for a specific time is another solution for the problem of dynamic modification of a node's behaviour. Every node in the black list has a specific time (termed as the isolated time) in which the node (say node m) is regarded as a malicious node by the owner (say node n) of the blacklist. During the isolated time, node m is insulated from forwarding packets. After the time, node m will be removed from the blacklist and its trust will be set to the black-list trust threshold. Node m will get a chance if node n has a packet to forward. If the packet is forwarded correctly by node m, the trust will increase. If the packet is not forwarded correctly, node m will be put into the blacklist again and be insulated for another term of the isolated time.

7 Conclusions

In this paper, we use packet forwarding ratio to evaluate a node trust and a continued product of node trusts to estimate a path trust. Combined with the simple model, a novel multipath reactive routing protocol (AOTDV) is proposed to discover trustworthy forward paths and alleviate the attacks from malicious nodes. In this protocol, a source establishes multiple trustworthy paths as candidates to a destination in a single route discovery. A route discovery is initiated only when all paths break or fail to meet the trust requirements of data packets. This protocol provides a flexible and feasible approach to choose a shortest path in all path candidates. Performance comparison of these routing protocols (AODV, AOMDV and AOTDV) shows that AOTDV is able to achieve a remarkable improvement in the packet delivery ratio and detect most malicious attacks.

For future work, we plan to incorporate the time-effect of forwarding ratio to the trust model, in which a forwarding ratio is divided into multiple windowed parts at intervals of a certain time. The forwarding ratio of a more recent window is given a larger weight. Moreover, we will consider an adaptive trust level classification of nodes taking into account the average trust value of all nodes. The problem of dynamic behaviour modification will also be considered. In addition, a comprehensive performance evaluation will be conducted to compare AOTDV with other routing protocols (e.g. DSR and TORA).

8 Acknowledgments

We would like to thank the reviewers of this paper for detailed and insightful comments. This research is sponsored by the Natural Science Foundation of China (NSFC) under grant nos. 60903031, 90718032, 60970003, the China Postdoctoral Science Foundation under grant no. 20090451310, the Postdoctoral Innovation Foundation of Shandong Province under grant no. 200902021 and the Independent Innovation Foundation of Shandong University under grant no. 2009TS032.

9 References

[1] JENSEN C.D., CONNELL P.O.: 'Trust-based route selection in dynamic source routing'. Proc. Int. Conf. on Trust Management, Pisa, Italy, May 2006, pp. 150–163

[2] ZAPATA M.G., ASOKAN N.: 'Secure ad hoc on-demand distance vector routing', *ACM Mob. Comput. Commun. Rev.*, 2002, **3**, (6), pp. 106–107

[3] HU Y.C., PERRIG A., JOHNSON D.B.: 'Ariadne: a secure on-demand routing protocol for ad hoc networks'. Proc. Int. Conf. Mobile Computing and Networking (Mobicom'02), Atlanta, Georgia, September 2002, pp. 12–23

[4] HU Y.C., PERRIG A.: 'A survey of secure wireless ad hoc routing', *IEEE Secur. Priv.*, 2004, **2**, (3), pp. 28–39

[5] GRIFFITHS N., JHUMKA A., DAWSON A., MYERS R.: 'A simple trust model for on-demand routing in mobile ad-hoc networks'. Proc. Int. Symp. on Intelligent Distributed Computing (IDC 2008), 2008, pp. 105–114

[6] GAMBETTA D.: 'Can we?', in GAMBETTA D. (ED.): 'Trust: making and breaking cooperative relations' (Oxford Press, 2000, 1st edn.), pp. 213–237

[7] PIRZADA A.A., MCDONALD C.: 'Trust establishment in pure ad-hoc networks', *Wirel. Pers. Commun.*, 2006, **37**, (1), pp. 39–168

[8] MARSH S.P.: 'Formalizing trust as a computational concept'. PhD thesis, Department of Mathematics and Computer Science, University of Stirling, 1994

[9] RESNICK P., ZECKHAUSER R.: 'Trust among strangers in Internet transactions: Empirical analysis of eBay's reputation system', in BAYE M. (ED.): (Elsevier Press, 2000, 1st edn.), 'Advances in applied microeconomics: the economics of the internet and E-commerce', pp. 127–157

[10] BUCHEGGER S., BOUDEC J.L.: 'A robust reputation system for p2p and mobile ad-hoc networks'. Proc. Int. Workshop on the Economics of Peer-to-Peer Systems, Cambridge, MA, USA, June 2004, pp. 119–123

[11] JØSANG A., ISMAIL R.: 'The beta reputation system'. Proc. 15th Bled Electronic Commerce Conf., Bled, Slovenia, June 2002, pp. 1–14

[12] SABATER J., SIERRA C.: 'Regret: reputation in gregarious societies'. Proc. Int. Conf. Autonomous Agents, Montreal, Canada, 2002, pp. 194–195

[13] SRIVATSA M., LIU L.: 'Securing decentralized reputation management using trustguard', *J. Parallel Distrib. Comput.*, 2006, **66**, (9), pp. 1217–1232

[14] ZHOU R., HWANG K.: 'Powertrust: A robust and scalable reputation system for trusted peer-to-peer computing', *IEEE Trans. Parallel Distrib. Syst.*, 2007, **18**, (4), pp. 460–473

[15] JIA Z., QIN Z., XU X., ZHANG R.: 'Depth-expurgation based dynamic trust evaluation algorithm for ad hoc networks'. Proc. Int. Conf. on Embedded Software and Systems, Sichuan, China, July 2008, pp. 399–404

[16] SELÇUK A.A., UZUN E., PARIENTE M.R.: 'A reputation-based trust management system for P2P networks'. Proc. Int. Symp. on Cluster Computing and the Grid, Chicago, USA, April 2004, pp. 251–258

[17] XIONG L., LIU L.: 'PeerTrust: Supporting reputation-based trust in peer-to-peer communities', *IEEE Trans. Knowl. Data Eng.*, 2004, **16**, (7), pp. 843–857

[18] SONG S., HWANG K., ZHOU R., KWOK Y.-K.: 'Trusted P2P transactions with fuzzy reputation aggregation', *IEEE Internet Comput.*, 2005, **9**, (6), pp. 24–34

[19] LIANG Z., SHI W.: 'Analysis of ratings on trust inference in open environments', *Perform. Eval.*, 2008, **65**, (2), pp. 99–128

[20] ROYER E.M., TOH C.K.: 'A review of current routing protocols for ad hoc mobile wireless networks', *IEEE Pers. Commun. Mag.*, 1999, **6**, (2), pp. 46–55

[21] PERKINS C.E., ROYER E.M., DAS S.R.: 'Ad-hoc on-demand distance vector routing'. Proc. Int. Workshop on Mobile Computing Systems and Applications (WMCSA), New Orleans, LA, USA., February 1999, pp. 90–100

[22] PERKINS C.E., BHAGWAT P.: 'Highly dynamic destination sequenced distance-vector routing (DSDV) for mobile Computers'. Proc. Int. Conf. ACM SIGCOMM, London, August 1994, pp. 234–244

[23] GARCIA-LUNA-ACEVES J.J., MOSKO M., PERKINS C.E.: 'A new approach to on-demand loop-free routing in networks using sequence number', *Comput. Netw.*, 2006, **50**, (10), pp. 1599–1615

[24] JOHNSON D., MALTZ D.: 'Dynamic source routing in Ad hoc wireless networks', in TOMASZ I., HANK K. (EDS.): 'Mobile computing', (Kluwer Academic Press, 1996, 1st edn.), pp. 153–181

[25] MARINA M.K., DAS S.R.: 'On-demand multipath distance vector routing for ad hoc networks'. Proc. Int. Conf. on Network Protocols, Riverside, CA, USA., November 2001, pp. 11–14

[26] PIRZADA A.A., MCDONALD C., DATTA A.: 'Performance comparison of trust-based reactive routing protocols', *IEEE Trans. Mob. Comput.*, 2006, **5**, (6), pp. 695–710

[27] MARTI S., GIULI T., LAI K., BAKER M.: 'Mitigating routing misbehavior in mobile ad hoc networks'. Proc. Int. Conf. Mobile Computing and Networking (MobiCom), Boston, MA, USA, August 2000, pp. 255–265

[28] PIRZADA A.A., DATTA A., MCDONALD C.: 'Propagating trust in ad hoc networks for reliable routing'. Proc. Int. Workshop Wireless Ad Hoc Networks (IWWAN), Oulu, Finland, May 2004, pp. 58–62

[29] SUN Y., YU W., HAN Z., LIU K.J.R.: 'Information theoretic framework of trust modeling and evaluation for ad hoc networks', *IEEE J. Sel. Areas Commun.*, 2006, **24**, (2), pp. 305–317

[30] NASIPURI A., CASTANEDA R., DAS S.R.: 'Performance of multipath routing for on-demand protocols in mobile ad hoc networks', *Mob. Netw. Appl.*, 2001, **6**, (4), pp. 339–349

[31] http://www.isi.edu/nsnam/ns/, accessed September, 2009

[32] 802.11: 'Wireless LAN medium access control (MAC) and physical layer (PHY) specifications 802.11', 1997

[33] TANENBAUM A.S.: 'The medium access control sublayer', in TANENBAUM A.S. (ED.): 'Computer networks' (Prentice Hall Press, 2002, 4th edn.), pp. 251–270

[34] BETTSTETTER C., RESTA G., SANTI P.: 'The node distribution of the random waypoint mobility model for wireless ad hoc networks', *IEEE Trans. Mob. Comput.*, 2003, **2**, (3), pp. 257–269

[35] DOUCEUR J.R.: 'The Sybil Attack'. Proc. Int. Workshop on Peer-to-Peer Systems, Cambridge, MA, USA, March 2002, pp. 251–260

[36] PIRZADA A.A., MCDONALD C.: 'Reliable routing in ad hoc networks using direct trust mechanisms', in CHENG X.Y., LI D.Y. (EDS.): 'Advances in ad hoc and sensor networks' (Springer Press, 2006, 1st edn.), pp. 122–160

[37] CHENG W., LIAO X., SHEN C., LI S., PENG S.: 'A trust-based routing framework in energy-constrained wireless sensor networks'. Proc. Int. Conf. on Wireless Algorithms, Systems, and Applications (WASA), Xi'an, China, August 2006, pp. 478–489

10 Appendix

Theorem 1: The route update rule in AOTDV yields loop free routes.

Proof: By contradiction.

Suppose that there is an m-sized loop in a route to a destination d, and node links $(i_1 \to i_2 \to \cdots \to i_m \to i_1)$ form in a loop route. Assume that nodes j and k are any two consecutive nodes in the route path (j,k $\{i_1, i_2, \ldots, i_m, i_1\}$) and j is the previous node of k (i.e. $j \to k$). The following condition holds in lines 1 and 8 (discussed in Section 4.6)

$$\text{SeqNumber}_j^d \le \text{SeqNumber}_k^d$$

The following inequality must be true in the path $i_1 \to i_2 \to \cdots \to i_m \to i_1$.

$$\begin{aligned}\text{SeqNumber}_{i1}^d \le \text{SeqNumber}_{i2}^d \le \cdots \le \text{SeqNumber}_{im}^d \\ \le \text{SeqNumber}_{i1}^d\end{aligned}$$

which implies $\text{SeqNumber}_{i1}^d = \text{SeqNumber}_{i2}^d = \cdots = \text{SeqNumber}_{im}^d = \text{SeqNumber}_{i1}^d$. In other words, the destination sequence numbers are the same for every node in the routing loop.

The following condition holds in line 9

$$
\begin{aligned}
&(\mathrm{MaxTrust}_{i1}^{d}, -\mathrm{MinHops}_{i1}^{d}) < (\mathrm{MaxTrust}_{i2}^{d}, -\mathrm{MinHops}_{i2}^{d}) \\
&\quad < \cdots < (\mathrm{MaxTrust}_{im}^{d}, -\mathrm{MinHops}_{im}^{d}) \\
&\quad < (\mathrm{MaxTrust}_{i1}^{d}, -\mathrm{MinHops}_{i1}^{d}) \qquad (4)
\end{aligned}
$$

where $(\mathrm{MaxTrust}_{j}^{d}, -\mathrm{MinHops}_{j}^{d}) < (\mathrm{MaxTrust}_{k}^{d}, -\mathrm{MinHops}_{k}^{d})$ iff

a. $(\mathrm{MaxTrust}_{j}^{d} < \mathrm{MinTrust}_{k}^{d})$, or

b. $(\mathrm{MaxTrust}_{j}^{d} = \mathrm{MinHops}_{k}^{d})$ and $(-\mathrm{MinHops}_{ij}^{d} < -\mathrm{MinHops}_{k}^{d})$.

From inequality (4), we obtain

$$
(\mathrm{MaxTrust}_{i1}^{d}, -\mathrm{MinHops}_{i1}^{d}) < (\mathrm{MaxTrust}_{i1}^{d}, -\mathrm{MinHops}_{i1}^{d}) \qquad (5)
$$

Inequality (5) clearly is impossible; this means that node links $(i_1 \rightarrow i_2 \rightarrow \cdots \rightarrow i_m \rightarrow i_1)$ does not exist.

Thus, routes formed by AOTDV are loop free. □