STACRP: A secure trusted auction oriented clustering based routing protocol for MANET

Article *in* Cluster Computing · September 2012

DOI: 10.1007/s10586-012-0198-7

CITATIONS
31

READS
244

3 authors:

Pushpita Chatterjee
SRM Institute of Science and Technology
41 PUBLICATIONS **319** CITATIONS
SEE PROFILE

Indranil Sengupta
Indian Institute of Technology Kharagpur
142 PUBLICATIONS **1,763** CITATIONS
SEE PROFILE

Soumya K. Ghosh
Indian Institute of Technology Kharagpur
305 PUBLICATIONS **4,782** CITATIONS
SEE PROFILE

Cluster Comput (2012) 15:303–320
DOI 10.1007/s10586-012-0198-7

STACRP: a secure trusted auction oriented clustering based routing protocol for MANET

Pushpita Chatterjee · Indranil Sengupta · S.K. Ghosh

Received: 15 May 2011 / Accepted: 4 January 2012 / Published online: 19 January 2012

Abstract In mobile ad hoc network (MANET) nodes have a tendency to drop others' packet to conserve its own energy. If most of the nodes in a network start to behave in this way, either a portion of the network would be isolated or total network functionality would be hampered. This behavior is known as selfishness. Therefore, selfishness mitigation and enforcing cooperation between nodes is very important to increase the availability of nodes and overall throughput and to achieve the robustness of the network. Both credit and reputation based mechanisms are used to attract nodes to forward others' packets. In light of this, we propose a game theoretic routing model, Secure Trusted Auction oriented Clustering based Routing Protocol (STACRP), to provide trusted framework for MANET. Two auction mechanisms procurement and Dutch are used to determine the forwarding cost-per-hop for intermediate nodes. Our model is lightweight in terms of computational and communication requirements, yet powerful in terms of flexibility in managing trust between nodes of heterogeneous deployments. It manages trust locally with minimal overhead in terms of extra messages. STACRP organizes the network into 1-hop disjoint clusters and elects the most qualified and trustworthy nodes as Clusterhead. The trust is quantified with carefully chosen parameters having deep impact on network functionality. The trust model is analyzed using Markov chain and is proven as continuous time Markov chain. The security analysis of the model is analyzed to guarantee that the proposed approach achieves a secure reliable routing solution for MANETs. The proposed model have been evaluated with a set of simulations that show STACRP detects selfish nodes and enforces cooperation between nodes and achieves better throughput and packet delivery ratio with lees routing overhead compare to AODV.

Keywords Cluster · Security · Trust · Attack · Routing

1 Introduction

Mobile Ad Hoc Networks (MANET) are formed by a decentralized distributed collection of wireless mobile nodes having dynamic topology, multi-hop routing, node mobility, no fixed routing protocol without any predesigned infrastructure. They are characterized by randomly changing topologies, distributed control and cooperative behavior. In order to make an ad hoc network functional, the nodes are assumed to follow a self-organizing protocol, and the intermediate nodes are expected to relay messages between two distant nodes. Applications of mobile ad hoc networks have been envisioned mainly for emergency and military situations and it may be extended for civilian applications as well.

The proposed scheme consists of a trust based secure clustering framework and various attack analysis in distributed realm. Security always comes with extra cost in the form of computation power, communication overhead, latency etc. Therefore, tradeoff between security and overhead is needed. Selfishness of mobile nodes is an important area of MANET research. The mobile nodes may have a tendency to drop others' packet with the aim to conserve

P. Chatterjee (✉) · S.K. Ghosh
School of Information Technology, Indian Institute of Technology, Kharagpur, India
e-mail: pushpitac@sit.iitkgp.ernet.in

S.K. Ghosh
e-mail: skg@iitkgp.ac.in

I. Sengupta
Dept. of Computer Science & Engineering, Indian Institute of Technology, Kharagpur, India
e-mail: isg@iitkgp.ac.in

its energy by going to sleep mode while it has no packet to send. As a result the packet may not reach its intended destination. The problem occurs, when most of the nodes in a network start to behave in this way, either a portion of the network would be isolated or total network functionality would be hampered. This behavior is known as *selfishness* of the nodes. To mitigate selfishness and enforcing cooperation between the nodes, a rewarding scheme is proposed for good behavior towards proper functionality of the network. In recent years trust and ubiquitous computing has made vast advancements. Several trust oriented routing models have been proposed for secure routing in MANET. Nodes must be prepared to operate in a mode that should not immediately trust on any peer. In MANET, the absence of fixed infrastructure, limited resources, ephemeral connectivity and availability, shared wireless medium and physical vulnerability, makes trust establishment very complex. The trust established between network nodes could be used for the provision of higher level security solutions, such as trusted key exchange or secure routing. The existence of trusted third parties used as intermediaries for establishing trust relationships cannot be taken for granted, trust relationships change frequently due to the dynamic topology. Moreover trust should not be transitive. Improving the cooperation of the nodes may increase the available bandwidth of the network, since the increased connectivity of the network leads to more possible routes in the network. If more number of nodes are cooperating, the average number of packets that each node has to forward will be reduced, which leads to lower energy consumption and more fairness in the network. Without node cooperation, route can be hardly established, packet forwarding would be hampered. The uncooperative mobile nodes can be broadly classified into two categories: *Faulty or malicious*—Either they cannot follow a protocol or are intentionally malicious and try to attack the system; *Selfish*—The objective of selfish nodes is to maximize its own welfare (to save battery for its own communication), which is defined as the benefit from their action minus the cost of the action.

In this paper, we propose a Secure Trusted Auction oriented Clustering based Routing Protocol (STACRP) that can mitigate selfishness and enforce cooperation of nodes using both incentive and punishment approach in MANET. To determine the optimum cost of packet forwarding per hop basis, two types of auction, *Dutch* and *Procurement* mechanisms are used. Trust is used as a metric to determine cost for forwarding. If a node is trusted and good, it needs very low cost to pay to send its own packet. Also there is a chance for selfish node to recover its status to become *good* by cooperation. Thus we can attract nodes to cooperate in the network for its own benefit. To the best of our knowledge, this is the first endeavor to incorporate trust rating as a metric for forwarding a packet. In addition to this, the proposed model is analyzed using Markov model to evaluate its performance.

The rest of the paper is organized as follows: Sect. 2 gives a brief note on the state of the art of the research efforts in the area of approaches (both rewarding and punishing) for selfishness mitigation and trust management issues in the realm of distributed networking. The proposed model is briefly introduced in Sect. 3. Section 4 analyzes the proposed trust model with Markov model followed by Sect. 5 that analyzes the protocol with different attacks and shows the robustness of the protocol and Sect. 6 describes the simulation environment, parameters, performance metrics and results and finally, conclusions are given in Sect. 7.

2 State of the art

The cooperation enforcement techniques play important role in MANET, when the primary goal is the availability, the robustness of the network, and the overall throughput. These techniques are categorized as credit-based and reputation based. The former category is based on economic incentives (pricing or credit-based) and the second is based on reputation building to enforce cooperation.

2.1 Rewarding/credit based scheme

Proper packet forwarding task, is treated as a service for the reward models, that can be evaluated and charged. These models incorporate a form of virtual currency to regulate the dealings between the various nodes for packet forwarding. They require the existence of tamper resistant hardware or a virtual bank. In [1], Buttyan and Hubaux proposed a cooperation stimulation approach based on a virtual currency, called *nuglets*, which are used as payments for packet forwarding. In packet purse model (PPM), the sender of a packet pays by loading some *nuglets* in the packet before sending it. Intermediate nodes acquire some *nuglets*, from the packet when they forward it. If the packet runs out of *nuglets*, then it is dropped. In packet trade model (PTM), the destination of a packet pays for the packet. To implement either PPM or PTM, a tamper-proof hardware is required at each node to ensure that the correct amount of *nuglets*, is deducted or credited at each node. There are some other issues also exist for PPM and PTM: Both models require the clearance of nuglets, in real-time. As a result, if the system does not have enough *nuglets*, circulating around, the performance of their system may degrade. Under both models, if a mobile node runs out of nuglets, its tamper-proof hardware still has to contact with some central authority in order to *refill* its credit. Actually, the Central Bank used by our system is similar to such an authority. A disadvantage of PTM is that it is vulnerable to network overload, since the senders do not have to pay. Sprite [2] was proposed by Zhong et al. It does not require tamper-proof hardware to

prevent the deviation of payment units, but incorporates a centralized credit clearance service (CCS). Sprite provides integrity during packet exchanges, and is based on digital signatures. Another scheme, introduced by Yang et al. [3], protects both routing and packet forwarding in the context of the AODV [4]. It is self-organized, without assuming any a-priori trust between the nodes or the existence of any centralized trust entity. It isolates the misbehaving nodes and employs threshold cryptography to enhance the tolerance against these nodes. The scheme is fully localized (one hop), and its credit based strategy produces overhead that is significantly decreased when the network is not harmed. Anderegg et al. proposed a scheme [5] using VCG, second-best sealed type of auction, that works on top of the dynamic source routing (DSR) [6]. This estimates the cost may be asked by a node to forward others' packet through the *cost-of-energy* parameter. However, there is a limitation: the nodes have to indicate the signal strength at which they emit and they also need to forward information regarding their neighbors received signal strengths. This provides means for nodes to cheat. Though the author proved that it is not profitable for a node to alter the cost-of-energy parameter, because the final cost is higher than the extra profit the node makes. The ad hoc-VCG is robust when only one cheating node exists. It might fail in the presence of collusion of nodes who try to maximize their payments.

2.2 Reputation scheme

The reputation-based models use reputation of nodes to forward packets through the most reliable nodes. The reputation of a node increases when it carries out rightly the task of forwarding the packets that are dispatched by its neighbors, without altering their fields. Recently, several selfishness mitigation and cooperation enforcement mechanisms have been proposed in [7–13]. Buchegger et al., proposed CONFIDANT scheme [10] and this scheme facilitates monitoring and reporting for a route establishment that avoids the misbehaving nodes in DSR. The first version of CONFIDANT was vulnerable to rumor spreading phenomena. Some enhancement is proposed based on Bayesian model. However, it does not use tamper-proof hardware. For a misbehaving node, it is hard to know the entries of its reputation in other nodes or to modify its reputations. Michiardi and Molva proposed CORE scheme [11] based on DSR. It enforces node collaboration through monitoring of the cooperativeness of nodes and a reputation mechanism. The scheme is immune to attacks performed using the mechanism itself: no negative ratings are spread, and thus, it is impossible for a node to maliciously decrease another nodes reputation. Also, it does not discriminate malfunction and misbehaving nodes. Additionally, a second chance mechanism is not consolidated, as in OCEAN scheme [12]. Hence, a malfunctioning node can not rebuild its reputation when it recovers from temporal problems. OCEAN [12] is a hybrid scheme that uses both a reputation based component to detect and punish selfish behavior, and a micro-payment component to encourage cooperation. The credit is earned for each immediate neighbor and it cannot be used to send packets in a different route. Milan et al. proposed a scheme [13], where a game-theoretic model to study the impact of collisions on a hop-by-hop reputation-based mechanism for regular networks with uniform random traffic. The devices in MANET are equipped with different resources and provide discrete services, it did not deal with irregular topologies and non-uniform routing, that will introduce perception and interaction asymmetries that could impair cooperation.

2.3 Handling trust management issues

Several trust models [14–19], have been proposed for self organized networks in distributed paradigm. Pirzada et al. [20] proposed and examined the efficiency of trust-based reactive routing protocols in the presence of attacks. This work only considers first hand information to evaluate other nodes trust values. Pisinou et al. [21] proposed a secure AODV-based routing protocol for multi-hop ad hoc networks to establish a secure end-to-end route. The trust values are calculated based only on direct observations, assuming that trust is transitive. Ghosh et al. [22] enhanced trust management by considering the confidence level of trust. They have used the confidence level as a weight on the computed trust value. Balakrishnnan et al. [23] developed a trust model to strengthen the security of MANETs and to deal with the issues associated with recommendations. Their model utilizes only trusted routes for communication, and isolates malicious nodes based on the evidence obtained from direct interactions and recommendations. Wang et al. [24] proposed a mechanism extension to AODV, to distinguish selfish peers from cooperative ones based solely on local monitoring. In order to distinguish between selfish and cooperative peers, a series of well-known statistical tests are applied to features obtained from the observed AODV actions. Virendra et al. [25] proposes a technique for quantifying trust depending upon some metrics, which have good impact network functionalities. Chang and Kou [26] proposed a Markov chain trust model to obtain the trust value (TV) for 1-hop neighbors. However, it computes TVs only based on direct observations and does not consider trust decay due to using recommendations from remote nodes [27].

3 STACRP—the proposed protocol

The contribution of the proposed secure trusted auction oriented clustering based routing protocol (STACRP) are as follows:

– Subdivide the network into several small groups known as *clusters*, nodes are grouped based on their geographic location,
– establish trust relationship between nodes to ensure secure end-to-end delivery of data and prevent some attacks,
– and enforcing cooperation between nodes to mitigate selfishness and increase the availability.

3.1 Preliminaries

Following are the assumptions made in STACRP:

– nodes are identical in their physical characteristics, that is, if a node A is within the transmission range of node B then node B is also within the transmission range of node A;
– nodes communicate via a shared bi-directional channel;
– nodes operate in promiscuous mode;
– initially uniform trust value is assigned to each node;
– initially same amount of virtual currency (*VC*) is loaded in all nodes;
– nodes are equipped with a residual energy detection software.

We broadly categorizes nodes into three groups: member node (MN), clusterhead (CH) and guard node (GN). MNs are ordinary nodes in a cluster, which send, receive and route packets and monitor neighbor nodes promiscuously and calculate direct trust evidence. CH is the cluster master responsible for trust and reputation calculation and certificate distribution, routing initialization, session key generation for secure data delivery, route maintenance, packet forwarding cost distribution and removing bad behaving node from cluster. GN is responsible for monitoring the CH. As the sole clustering framework depends on the performance of the CH, CH is needed to be observed whether it behaves truthfully or not. It is obvious that all monitoring and evaluation work costs energy, to prolong the node lifetime and the network as a whole, is an important issue. All trusted member nodes have to act as GN for a certain period of time, for this we formalize a round-robin like scheduling mechanism for GN and CH. Detail of this proposal is beyond the scope of this paper.

3.2 System model

System model of the proposed STACRP is presented in Fig. 1 and its components are as follows:

– *monitor module:* the main function is to monitor the packet transmissions and collect useful data about network functionalities, such as packet forwarding, dropping, misrouting, false route injection. MN monitors each neighbor MN in short-term monitoring. However, if a node finds any MN is malfunctioning, it monitors the particular misbehaved node for long term;
– *processing module:* this module compares collected data with predefined threshold and find deviation;
– *trust module:* getting the deviation from previous module, this module calculates direct trust and combines direct evidence with collected recommendation from its neighbors;
– reputation module this module calculates the selfishness of any node using reputation value *RV* and allowable threshold;
– *decision module:* after calculating resultant-trust, this module is called to categorize nodes according to their trust and selfishness metric;
– *routing module:* this takes all routing decision according to the status of the source and destination nodes.

3.3 Clustering framework

To start with, we first introduce the proposed clustering scheme in brief. The proposed clustering technique uses both hop distance and number of member nodes in a cluster to control the cluster structure. The most trustworthy node in each cluster takes the role of the clusterhead (CH). CH is elected using a secret voting scheme. The proposed clustering technique organizes the nodes within one hop from a node called CH into a cluster. When a node moves away from its CH and the distance between the node and its CH is larger than 1, it joins a new cluster if it can find an existing CH within 1 hop; otherwise, a new cluster is formed with the node as the CH. When the distance between two clusterheads is detected to be less than or equal to a predetermined number of hops, D (here $D = 1$), the cluster, with less number of member nodes, is recycled. Each of the nodes in the recycled cluster finds a new cluster to join. The parameter D is referred to as the cluster-recycle distance. For clustering purpose, each node maintains a small amount of information of itself and its neighboring nodes. Let the cluster information of a node with ID i be represented by a 7-tuple (pubkey, node-id, cluster-id, CH-id, trust, SOC, neigh) where the seven fields of the 7-tuple correspond to the seven fields described previously. For a CH node id and CH id is same. When a node, say i, is powered up, it sets its cluster information to be (pubkey, i, 0, 0, 1, SOC, 0) indicating that it is in its *initialization* phase. It then searches for its neighboring nodes. If there is at least one neighboring node, it sends its cluster information to all of its neighboring nodes. Upon receiving the cluster information of node i, each neighbor node adds node i as one of its neighbors. Each neighboring node which detects the existence of node i also sends its cluster information to node i. When node i receives the cluster information from all of its neighboring nodes, it checks whether there is at least one of its neighboring nodes which is not in the *initialization* phase (to ensure it is not the first node in the network or become *standalone* node). If

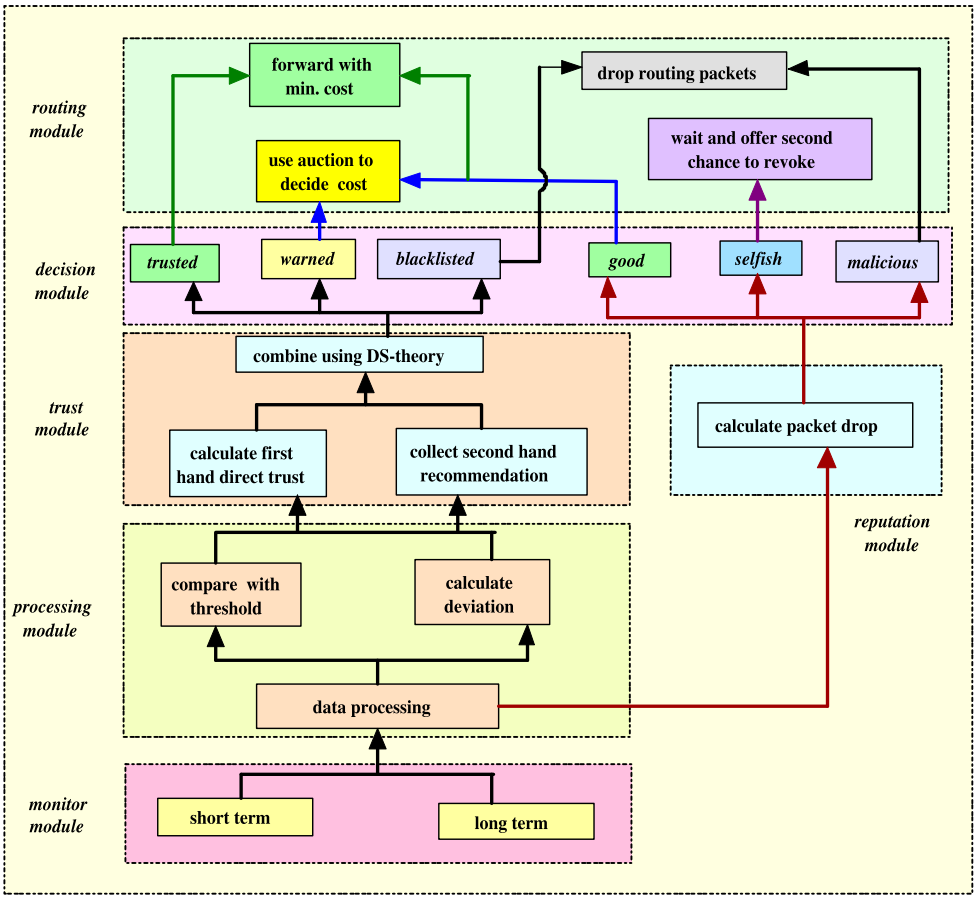

Fig. 1 System model used in STACRP

yes, it tries to find is any of the replying node is CH. If such a cluster is found, it joins the cluster and updates its cluster information. It then sends its clustering information to all of its neighboring nodes. Upon receiving the cluster information of node *i*, each neighboring node updates the information maintained by it. This is known as *initial clustering*. After certain round of communication clusterhead election is revoked. The detail discussion on cluster setup, maintenance, latency and overhead and load balancing are beyond the scope of the paper.

3.4 Trust and reputation modeling

As trust value is subject to review, after a certain time, each node evaluates the trustworthiness of its one hop neighbor nodes. We carefully choose certain parameters to quantify the trust of a node according to its contribution towards proper functioning of the network and minimizing the number of bad nodes from the network. Another important aspect considered in this work that is stimulating cooperation between the nodes. The behavior of a node which play the important role in formalizing trust are: packet forward, packet drop, packet misroute and packet false injection. The direct trust is calculated using this procedure:

Step 1 A node monitor/collect values of trust parameters and calculates the count of each behavior.
Step 2 Calculates deviation from predefined threshold for each parameter.
Step 3 Calculate the corresponding direct trust value.

Thresholds are set according to application scenario. The resultant trust is determined by combining self evidence of CH and recommendation evidences collected from 1-hop neighbors of node under review. The evidences are combined using weighted Dempster-Shafer mathematical model. The vivid description of trust calculation is described in [28]. The overall description is beyond the scope of this paper.

As mitigating selfishness of the mobile hosts is the main focus of the present work, the reputation module is discussed vividly. To determine the reputation state of a node, packet drop count must be monitored carefully. The main components of the Reputation module are depicted in Fig. 2. The components of the *Reputation module* are as follows: The *Reputation module* works in the CH of a cluster. CH collects packet drop information from all 1-hop neighbors of the node under review. Subsequently, CH calculates the mean and standard variation of collected packet drop data. A max and min threshold can be set depending upon the application scenario. If the reputation value (*RV*) crosses

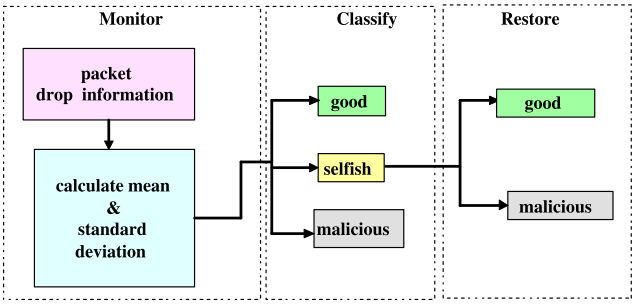

Fig. 2 Reputation module

the max threshold, the node is determined as *malicious*. The reputation-wise classification of nodes is given in Algorithm 1. The main functionalities of the *Reputation module* are

- *Monitor*: It monitors the traffic of a node to detect whether the node misbehaving or not.
- *Classify*: It classifies the nodes according to their *RV*.
- *Restore*: It does not isolate any misbehaving node right away, gives chances to selfish nodes to restore its status by behaving properly.

Algorithm 1: State of the node reputation basis (NodeState)

Input: Different reputation evidences about a node M
Output: Reputation state of M
Each neighbor N of M calculates the Packet_Received and Forward_Count RCV_M, FWD_M at any interval $t_2 - t_1$;
$RV_M^N = (RCV_M - FWD_M)/RCV_M$;
RV_M^N sends to CH;
CH calculates the mean μ;
$\mu = \frac{1}{n}\sum_{i=1}^{n} RV_M^i$; /* where n is total number of neighbors of M */;
CH calculates standard deviation σ;
$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(RV_M^i - \mu)^2}$;
$RV_{Min\text{-}Threshold} = \mu + k_1 * \sigma$;
$RV_{Max\text{-}Threshold} = \mu + k_2 * \sigma$;
$RV = \mu$;
if $RV < RV_{Min\text{-}Threshold}$ **then**
  M is *Good*;
end
if $RV_{Min\text{-}Threshold} > RV < RV_{Max\text{-}Threshold}$ **then**
  M is *Selfish*;
else
  M is *Malicious*;
end

Here, in our proposal, we consider 15–20% of total number of packets may drop due to different network conditions (e.g., high mobility of nodes, network congestion, obstacles in the medium, or interference of other networks), which cannot be directly determined. Therefore, we allow 20% packet drop as a good behavior and accordingly categorize the nodes as follows:

- *Trusted*—node having trust greater than 0.5.
- *Warned*—node with trust value lies between 0.25 and 0.49.
- *Blacklisted*—node with trust value less than 0.25.
- *Good*—node having *RV* less than Min_{th}.
- *Selfish*—node with *RV* lies between Min_{th} and Max_{th}.
- *Malicious*—node with *RV* greater than Max_{th}.

3.5 Timer description

Following are timers used in this scheme. It is to be noted that a down counter is kept along with each timer to determine the timer expiring time.

- $\tau_{Procurement}$—Source node wants to sell its packet starts this timer after broadcasting *Req-bid* message to one-hop neighbors. The timer stops when it is timed out. This is the waiting time to receive *Reply-bid* from interested bidder.
- $\tau_{Procurement\text{-}accept}$—This timer starts by an interested intermediate bidder to get *Accept-bid* from the source and timer stops when time is elapsed.
- τ_{Dutch}—This timer starts intermediate node after broadcasting *dutch-bid* and stops when timed out. This is the waiting time for receiving *Accept-dutch* message from interested buyer.

3.6 Clusterhead election

The clusterhead is a node which wins the CH election. Any initial CH or any node wants to be CH can initiate the CH election by broadcasting *Start-Elect* message. The brief CH election procedure is described below:

Step 1 A node *M* broadcasts *Start-Elect* message with residual battery power, mobility value and no. of neighbors (B_{RC}, M_V, N_R) to start the CH election.
Step 2 each neighbor node keeps T_V evidence of the initiator node. Each neighbor calculates *Elect_weight* if *M* is *trusted*.
Step 3 neighbor nodes vote for *M* if *Elect_weight* is greater than a predefined threshold. *Elect_weight* is a weighted combination of T_V, B_{RC}, M_V, and N_R, where weight to each parameter is assigned depending upon the application scenario.
Step 4 if *M* wins by getting at least *n/2* votes where *n* is the number of neighbors, *M* declares as CH.
Step 5 Voter nodes becomes cluster members.

3.7 Routing

In this work, we mainly focus on network layer and it does not assume that nodes have any knowledge about the network. A node has no idea about what other nodes can bid while they are asked for bidding. So, they try to bid as low

as possible to get the packet, because by forwarding a packet with a optimum cost the forwarding node gains trust and currency also. The packet forwarding cost calculated by our method is optimum one. Each node keeps a table of trust value of its all one hop neighbors. The protocol works on the top of AODV with bidding mechanism and following are the steps for routing:

3.7.1 Neighbor management

Each active node keeps track of its single hop neighbor nodes of a same cluster or of different clusters by exchanging periodic Hello messages. CH maintains a *Member table* for all its member nodes. The member table consists the member id, trust value, remaining battery capacity by exchanging information periodically.

3.7.2 Route discovery

A source node M_1, wants to communicate with destination node M_2, sends the *RREQ* to CH. CH checks the status of both M_1 and M_2. Depending on the status of said nodes, the following cases may occur:

Case 1: M_1 and M_2 trusted and good If CH finds M_1 and M_2 are *trusted* (having $T_V \geq 0.5$) and *good* ($RV \leq Min_{\text{th}}$), it initiates the route discovery phase. CH generates a session key *SK* and sends to them by encrypting ($E_{KS}\{SK\}$), ($E_{KD}\{SK\}$) with public key of M_1 and M_2 respectively. If M_1 and M_2 are 1-hop, no *VC* is charged which is shown in Fig. 3(a) and Fig. 3(b). However, for non-neighbor M_1 and M_2, packet forwarding cost is set to a minimum fixed per hop charge. Each intermediate node has to forward the packet at this cost. Figure 3(c) and Fig. 3(d) depicts the route establishment when both M_1 and M_2 are non neighbor, trusted and good.

Case 2: M_1 and M_2 trusted but selfish While CH finds M_1 and M_2 nodes are *Trusted* (having $T_V \geq 0.5$) but *selfish* ($0.2 > RV < 0.3$), then CH generates WAIT signal to corresponding node. Figure 3(e) presents the route establishment procedure for selfish nodes. If M_2 is *selfish* then CH informs M_1 to wait for certain time so that M_2 can revive its status. If the *selfish* node continues to misbehave, CH isolates the node.

Case 3: M_1 and M_2 warned but good While CH finds M_1 and M_2 are *warned* (having $0.25 < T_V < 0.5$) but *good* ($RV \leq 0.2$), CH initiates route discovery phase. CH generates a session key *SK* and sends to them by encrypting ($E_{KS}\{SK\}$), ($E_{KD}\{SK\}$) with public key of M_1 and M_2 respectively. Then CH broadcasts the RREQ. If M_1 and M_2 are not in 1-hop, two auctions namely, Procurement and Dutch are called to sell the packet and the total cost *VC* of the packet is debited equally from M_1 and M_2 which is briefly depicted in Fig. 3(f). Each intermediate forwarding node gets *VC* as described in Algorithm 3.

Case 4: M_1 and M_2 blacklisted and malicious If CH finds any or both of the M_1 and M_2 are *blacklisted* ($T_V < 0.25$) and *malicious* ($RV > 0.3$), CH drops the *Route Permit* request.

Each node forwarding a packet sends the information of auction price to CH to add the currency in its account. Getting full information and price (say K) from M_1 and M_2, the CH then pays the premiums to the intermediate nodes. The total cost of the packet is paid by both M_1 and M_2 equally ($K/2$), thus enforcing both M_1 and M_2 to behave truthfully and prevent generate false *RREQ* flooding. Algorithm 3 describes this procedure.

Our aim to is to enforce the cooperation between nodes. All nodes are equipped with some *VC*, the only way to earn *VC* by forwarding others' packets and each proper forwarding increases the trust level of the node. Once a node becomes *trusted*, it needs a minimum amount of *VC* to forward its own packets. So each node tries to be *trusted* and earn *VC* by forwarding others' packets otherwise gradually *VC* would be finished and due to less trust value the particular misbehaved node would be isolated from network. Thus proves our claim. In case of terminal nodes which has no option to forward others' packet, CH reloads *VC* to them to continue their functions. The overall route establishment procedure for all possible cases is described in Algorithm 2.

3.7.3 Route maintenance

This procedure explains how route problems (such as link breakage) are reported and recovered. Routes are maintained in the proposed scheme like AODV with following modification: as it forms 1-hop cluster, when an intermediate node detects the next hop is missing from the path, it unicasts RERR to the CH. A link may be broken either due to host mobility or due to selfishness. For the both cases, CH reinitiates the route discovery process to construct a new route, which reduces the routing overhead as compared to AODV.

4 Markov chain analysis of trust model

Markov analysis provides a means of analyzing the reliability and availability of systems whose components exhibit strong dependencies. The state transition diagram identifies all the discrete states of the system and the possible transitions between those states. In a Markov process the transition frequencies between states depends only on the current state probabilities and the constant transition rates between states. In this way the Markov model does not need to know about the history of how the state probabilities have evolved in time in order to calculate future state probabilities.Homogeneous Markov Chain is characterized by constant transition rates between the states. A memoryless system is characterized by the fact that the future state of the

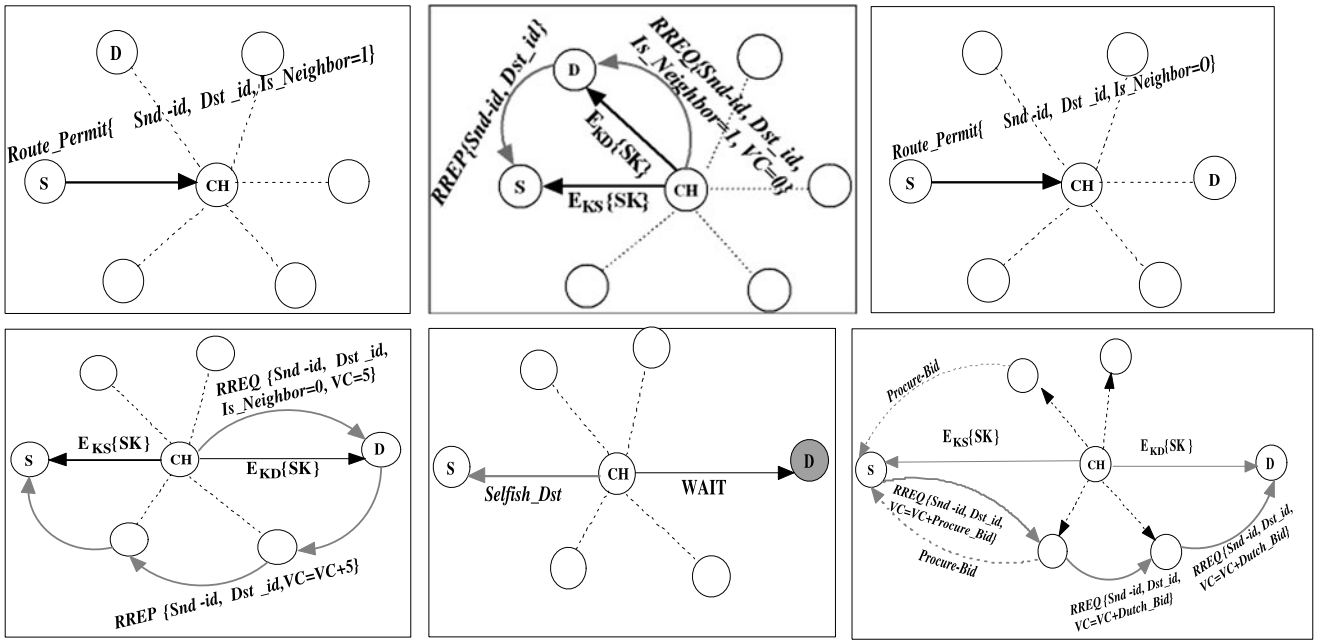

Fig. 3 (**a**) RREQ for *trusted* & *good* neighbor node; (**b**) RREP, SK for *trusted* & *good* neighbor node; (**c**) RREQ for non-neighbor *trusted* & *good* node; (**d**) RREP, SK for *trusted* & *good* non-neighbor node; (**e**) Route establishment for *trusted* but *selfish* node; (**f**) Route establishment for *trusted* but *warned* node

Table 1 Different trust events and corresponding change of trust state

Events	Good behavior		Bad behavior	
	Description	Change in state	Description	Change in state
Joining a cluster	Normal join	$S_i^T \rightarrow S_{i+1}^T$	Abnormal join	$S_i^T \rightarrow S_{i-1}^T$
Leaving a cluster	Graceful leave	$S_i^T \rightarrow S_{i+1}^T$	Graceless leave	$S_i^T \rightarrow S_{i-1}^T$
Cooperative/ non-coperative behavior	Packet forward	$S_i^T \rightarrow S_{i+2}^T$	Packet drop Misroute false Injection	$S_i^T \rightarrow S_{i-2}^T$ $S_i^T \rightarrow S_{i-1}^T$ $S_i^T \rightarrow S_{i-1}^T$
Revive from selfishness	From *selfish* to *good*	$S_i^T \rightarrow S_{i+2}^T$	From good to selfish	$S_i^T \rightarrow S_{i-2}^T$
Win CH election	Node wins election	$S_i^T \rightarrow S_{i+4}^T$	Existing CH looses	$S_i^T \rightarrow S_{i-4}^T$
Available energy	SOC becomes SOC_{High}	$S_i^T \rightarrow S_{i+1}^T$	SOC becomes SOC_{Low}	$S_i^T \rightarrow S_{i-1}^T$

system depends only on its present state. A stationary system is one in which the probabilities that govern the transitions from state to state remain constant with time. In other words, the probability of transitioning from some state i to another state j is the same regardless of the point in time that the transition occurs. In this section, we propose a trust model using Markov chain to predict the trust of nodes in a cluster. We use the direct and observed trust of a node to predict the trust performance of a node. Here, we formalize the trust model using Markov model and prove the proposed trust model is a finite, ergodic CTMC. First we define several trust events, trust state is changed depending upon these events.

4.1 Trust events

Different trust events and their impact on the change of trust state is discussed in Table 1. It is evident that for an event trust state increases and decreases with equal probability. No other state is reached with zero probability.

– *Joining a cluster*—If a non-member node sends *join* message to join a cluster with proper format, it is a normal join event. But if a node tries to join in different clusters by sending several *join* messages in a short period of time, it represents bad manner. For the joining event trust state is changed like this: $S_i^T \rightarrow S_{i+1}^T$ for good behavior, $S_i^T \rightarrow S_{i-1}^T$ for bad behavior.

- *Leaving a cluster*—If a node leaves a cluster by sending *leave* message to CH and CH sends *translocation_id* for the leaving node, then the node leaves cluster gracefully and it is treated as good behavior. If a node leaves a node without informing the CH, it is treated as graceless leave of a node and trust state transition is $S_i^T \rightarrow S_{i-1}^T$, and for graceful leave, it is $S_i^T \rightarrow S_{i+1}^T$.
- *Cooperative/Non-cooperative behavior:* If a node behaves truthfully and towards proper functionality of the network it is represented as cooperative behavior. The behavior of a node can be determined by parameters like packet forwarding, packet drop, packet misrouting, packet false injection. For proper packet forwarding change of trust state is $S_i^T \rightarrow S_{i+2}^T$ and packet drop $S_i^T \rightarrow S_{i-2}^T$, for packet misrouting $S_i^T \rightarrow S_{i-1}^T$ and false injection $S_i^T \rightarrow S_{i-1}^T$. As this proposal basically handles the selfishness mitigation of nodes, the malicious packet drop is assigned the highest priority.
- *Available Energy:* To continue proper functionality of network nodes with more energy gives more reliability for transmission of a packet. We define energy threshold for any node SOC_{Low} and SOC_{High}. If State of charge (SOC) of a node comes to greater or equal to SOC_{High} the trust state is $S_i^T \rightarrow S_{i+1}^T$ and goes lower than SOC_{Low}, the trust state becomes $S_i^T \rightarrow S_{i-1}^T$.
- *Revive from selfishness:* Due to resource constraints, if node starts to behave selfishly (with higher drop ratio), CH warns the node. A node is treated as good if it starts to behave normally after getting warned from CH. The trust state of a good node is increased from $S_i^T \rightarrow S_{i+2}^T$. If a good node with proper SOC starts to behave non-cooperative, trust state is reduced by 2, that is $S_i^T \rightarrow S_{i-2}^T$.
- *Revive from warned*: If a misbehaving node (with lower trust value) revives its behavior from warned to trusted by doing proper functionality, its trust state becomes $S_i^T \rightarrow S_{i+2}^T$, the opposite behavior reduces the trust state from $S_i^T \rightarrow S_{i-2}^T$.
- *Win CH election:* If any non-CH node wins the CH election its trust state is increased by 4, that is $S_i^T \rightarrow S_{i+4}^T$, and trust state is transited from $S_i^T \rightarrow S_{i-4}^T$ if any current CH looses.

4.2 Model property

Definition 1 *Trust state:* We define the lowest trust state as 0 and **I** is the highest trust state. The arrival rate of $\lambda_{i,i+k}$ and departure rate of $\mu_{i,i-k}$ at any state i where $0 \leq i \leq \mathbf{I}$ and $1 \leq k \leq 4$. The trust increasing rate of a state is mapped with the arrival rate of a state and conversely departure rate of Markov queue is represented by trust decreasing rate of state.

The trust state transition may occur at arbitrary instants of time not merely fixed, at discrete points, so the proposed trust model can be modeled using Continuous-Time Markov Chain (CTMC). Now we give a formal definition of our proposed trust-CTMC:

Algorithm 2: Intra Cluster Routing (INTRA-ROUTE-DIS)

```
Input: A network with multiple clusters
Output: Secure route between two nodes M1 and M2 in a same cluster
foreach M1 wants to communicate M2 do
    M1 checks its neighbor list;
    if M2 is 1-hop neighbor of M1 then
        is_neighbor = 1;
    else
        is_neighbor = 0;
    end
    M1 sends permit route to CH;
    CH checks the Trust of M1;
    if M1 is Trusted then
        CH checks the state of M1;
        if M1 is Good then
            CH sets the Trusted-Sender = 1 in RREQ;
            CH set the packet-cost-per-hop = 5;
            CH checks status of M2;
            if CH finds M2 Trusted then
                if M2 is Good then
                    if M1 & M2 have VC ≥ VC_Threshold then
                        CH sets the Trusted-Receiver = 1 in RREQ;
                        CH generates a SK and encrypt with public key of M1 and M2;
                        CH sends encrypted SK to M1 and M2;
                    else not enough VC
                        CH generates WAIT signal;
                    end
                else M2 is selfish
                    CH generates WAIT Signal and informs M1;
                end
            else M2 not Trusted
                drops RREQ and informs M1;
            end
            CH checks is_neighbor;
            if is_neighbor = 1 then
                CH sends RREQ to M2;
                M2 sends RREP to M1;
                No VC is deducted;
            else
                CH broadcasts RREQ with One-Hop-flag = 0;
                Call Non-Neighbor-Trusted( );
            end
        else M1 is selfish
            CH generates a WAIT signal to revive state of M1 from selfish to good;
        end
    else M1 is blacklisted
        CH drops the message;
    end
    else if M1 and M2 warned then
        Call non-neighbor-warned( );
    end
end
```

Function (Non-Neighbor-Trusted)

All nodes in the cluster get the RREQ;
M_i checks Trusted_Sender and Trusted_Receiver bit;
if *Trusted_Sender and Trusted_Receiver bit* $= 1$ **then**
  if *Any intermediate node* M_i *finds both* M_1 & M_2 *is* 1*-hop from it* **then**
    M_i sends the *RREP* to CH with 1*-Hop-flag* $= 1$;
    if M_i *is* Trusted **then**
      CH sets a packet cost;
      CH digitally sign the *RREP* and sends to M_1;
    else
      CH drops the *RREP*;
    end
  else M_i not common neighbor of M_1 & M_2
    M_i rebroadcasts *RREQ* until M_2 is reached;
  end
end

Function (Non-Neighbor-Warned)

M_1 1-hop broadcasts *Req-bid* message;
for *Any Mobile-node* M_i *neighbor of* M_1 *not neighbor of* M_2 **do**
  M_i sends *Reply-bid* message with procurement bid to buy the packet to M_1;
end
M_1 waits for certain time τ_1 to receive *Reply-bid* message from interested neighbors;
M_1 checks trust value of bidding neighbors M_i;
if M_i *is* Trusted **then**
  M_1 sorts all bid-value and stores in ascending order;
  M_1 chooses the Trusted Minimum-bidder say M_j;
  M_1 sends the *Accept-bid* packet to M_j;
else M_i *Blacklisted*
  Drops the request;
end
Getting *Accept-bid* packet M_i checks neighbor list;
if M_i *is 1-hop of* M_2 **then**
  M_i sends *RREP* to M_1;
else
  M_i sells the packet to next hop node by calling Dutch-Auction( );
end
M_1 waits for time τ_2 to receive *RREP* from M_2;
if *No* RREP *message comes from* M_2 **then**
  M_1 chooses the next minimum Trusted Bidder say M_k from sorted list;
  M_1 sends the *Accept-bid* packet to M_k;
end

Function (Dutch-Auction)

M_i broadcasts *dutch-bid*;
if *Any node* M_j *neighbor of* M_i *finds* M_2 *is* 1*-hop from it* **then**
  M_j checks the *dutch-bid*;
  if dutch-bid *is acceptable* **then**
    M_j sends the *Accept-dutch* to M_i;
  elsehigher *dutch-bid*
    M_j discards the message;
  end
else M_2 is not neighbor of M_j
  discard the *dutch-bid* message;
end
M_i waits for time τ_3;
if M_i *gets* Accept-dutch *from* M_j **then**
  M_i checks the trust value of M_j;
  if M_j *is Trusted* **then**
    Sends *RREQ* to M_j;
  else
    discards the message;
  end
else No *Accept-dutch* comes
  M_i reduces *dutch-bid* and rebroadcasts;
end

Algorithm 3: Packet Cost (Cost-Packet)

Input: Different Trust Parameters about a node M
Output: State of the Node
foreach *Member Node* **do**
  CH checks the trust value and state of M_1 and M_2;
  if M_1 *and* M_2 *are Trusted* **then**
    if M_1 *and* M_2 *are Good* **then**
      if M_1 *and* M_2 *is neighbor* **then**
        No *VC* is charged;
      else
        fixed-per-hop *VC* is charged;
      end
    else *selfish*
      WAIT to become *good*;
    end
  else *Warned*
    Packet cost is decided by auction mechanism;
  end
  else if *Malicious* **then**
    CH drops the request;
  end
end
Total packet cost *VC* is debited equally from M_1 and M_2;
VC is credited to each intermediate node;

Definition 2 A stochastic trust process $X_t : t \in T$ constitutes a discrete trust state CTMC model if for any arbitrary time $t_i \in R_0^+$ with $0 = t_0 < t_1 < t_2 < \cdots < t_n < t_{n+1}, \forall n \in N$ and any trust state $\forall s_i \in S = N_0$ for conditional probability mass function (*pmf*), the following relation holds:

$$P(X_{t_{n+1}} = s_{n+1} | X_{t_n} = s_n, X_{t_{n-1}} = s_{n-1}, \ldots X_{t_0} = s_0) = P(X_{t_{n+1}} = s_{n+1} | X_{t_n} = s_n) \quad (1)$$

where R_0^+ is subset of the set of non-negative real numbers is used to refer to the parameter set T of a CTMC.

Definition 3 *Memoryless and Time-homogeneous*—Trust behavior of our proposed model can be derived from the history. From the Markov model described above (1) current trust state can be derived only from the last trust state, so model holds memoryless property.

The (1) expresses the Markov property of the CTMC. Now we further analyze the time homogeneity of the proposed model. As the exponential distribution is only continuous distribution that provides memoryless property the state sojourn times must be exponentially distributed. Thus,

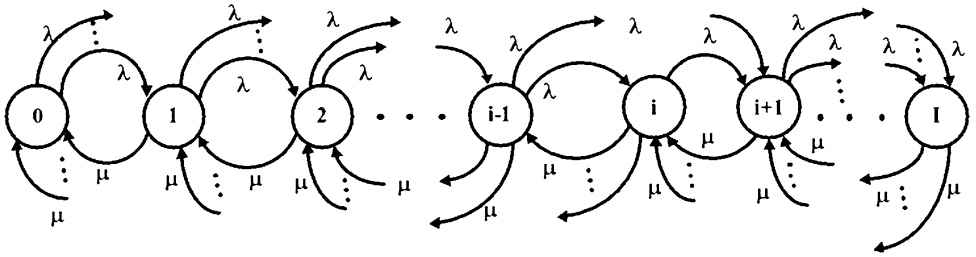

Fig. 4 Trust state transition diagram

the transition probability from state i to state j during a finite interval (t_1, t_2) with $t_1, t_2 \in T$ and $t_1 \leq t_2$ is given by the following (2):

$$p_{ij}(t_1, t_2) = P(X_{t_2} = j | X_{t_1} = i) \tag{2}$$

if $t_1 = t_2$

$$p_{ij}(t_1, t_1) = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

It is obvious that in our model the transition probability $p_{ij}(t_1, t_2)$ only depends on the time interval $t = t_2 - t_1$ not on the actual values of t_1 and t_2, therefore, transition probability of our model can be written as

$$\begin{aligned} p_{ij}(t) &= p_{ij}(0, t) = P(X_{t_1+t} = j | X_{t_1} = i) \\ &= P(X_t = j | X_0 = i), \quad \forall t_1 \in T \end{aligned} \tag{4}$$

Given the transition probability $p_{ij}(t_1, t_2)$ and the probability $\pi_i(t_1)$ of CTMC at time t_1, the unconditional state probability $\pi_j(t_2)$, $j \in S$ of the trust process at time t_2 can be easily derived. Using Chapman-Kolmogorov equation for the transition probabilities of a CTMC can be derived from (1) and by applying the theorem of total probability, we can write

$$\begin{aligned} &p_{ij}(t_1, t_2) = \sum_{k \in S} p_{ik}(t_1, t_3).p_{kj}(t_3, t_2), \\ &0 \leq t_1 \leq t_3 \leq t_2 \end{aligned} \tag{5}$$

Definition 4 *Transient*—Any trust state i is recurrent if the system will return to it some time in the future after leaving it. As our proposed model does not support recurrent trust state, it is transient.

Definition 5 *Irreducible*—As every trust state$i$ is reachable from every other trust state j, where $i, j \in S$; that is, $\forall i, j, i \neq j, \exists p_{ji}(t) > 0$. From analogy the proposed trust model is *irreducible*.

From Fig. 4, finite-state CTMC is represented by a state transition diagram, a finite directed graph, where state i of the chain is depicted by a vertex, and a one-step transition from state i to state j by an edge marked with one-step transition probability p_{ij}. A member node's trust state i is defined and determined depending upon the trust events i.e., normal/abnormal join, graceful/graceless leave, winning CH election, available battery power and other events described in Table 1. In real scenario, the above mentioned trust events on which trust state depends, may not follow the periodic feature. If a node monitors another node periodically with certain time interval any intellectual malicious node can easily deceive the monitoring node by behaving good only at that time. Therefore, trust evaluation must be aperiodic. Other environment factors like bandwidth, battery power, are also aperiodically changed.

Definition 6 *Finite and Aperiodic*—From the above discussion it is evident that if any trust state i of the trust CTMC is aperiodic, thus other states $j \in S$ also aperiodic. Consequently, all trust states of the trust CTMC are aperiodic and finite.

We define transition rate q_{ij} of the proposed trust CTMC from state i to state j using continuous time transition probability

$$q_{ij}(t) = \lim_{\Delta t \to 0} \frac{p_{ij}(t, t + \Delta t)}{\Delta t} \tag{6}$$

$$q_{ii}(t) = \lim_{\Delta t \to 0} \frac{p_{ii}(t, t + \Delta t) - 1}{\Delta t} \tag{7}$$

From (6) and (7) it is clear that since $p_{ij}(t, t + \Delta t) = 1$, at any instant of time t

$$\sum_{j \in S} q_{ij}(t) = 0, \quad \forall i \in S \tag{8}$$

In (7), $q_{ii}(t)$ interprets the total rate the trust model exited to other state at time t and accordingly, $q_{ij}(t), i \neq j$ denotes the total rate the trust model leaves state i in order to transit other state j at time t. As our proposed model is time homogeneous Markov chain, assuming $t = t_2 - t_1$, and using time independent transition rate q_{ij}, we can derive system of differential equations

$$\frac{d\pi_j(t)}{dt} = \sum_{i \in S} q_{ij} \pi_i(t), \quad \forall j \in S \tag{9}$$

Having these definitions and using *Chapman-Kolmogorov equation* we can define infinitesimal generator matrix $\mathbf{Q}$ of the transition probability matrix $\mathbf{P}(t) = [p_{ij}(0,t)] = p_{ij}(t)$ by refereing (6) and (7). The matrix $\mathbf{Q}$:

$$\mathbf{Q} = [q_{ij}], \quad \forall j \in S \tag{10}$$

Equation (10) contains the transition rate q_{ij} from any state i to other state j where $i \neq j$ of the trust Markov chain. Using the definition of (10), (9) can be rewritten as in vector matrix form

$$\dot{\pi} = \frac{d\pi_j(t)}{dt} = \pi(t)\mathbf{Q} \tag{11}$$

From the above analysis, we can write for all trust states $i \in S$, the steady trust state probabilities π_i are

- independent of time t,
- independent of initial trust state probability vector in other words, irrespective of the choice of initial trust state probability vector,
- the steady state probability will converge, the trust CTMC is ergodic,
- strictly positive,
- given as the time limits, $\pi_i = \lim_{t\to\infty}$ and $pi_i(t) = p_{ji}(t)$ respectively.

From the above property, the steady state probability of proposed trust CTMC are time independent and we can write

$$\lim_{t\to\infty} \frac{d\pi(t)}{dt} = 0 \tag{12}$$

Under this condition the unconditional steady state probability resolves to

$$0 = \sum_{i\in S} q_{ij}\pi_i(t), \quad \forall j \in S \tag{13}$$

in vector matrix form , we get

$$0 = \pi.\mathbf{Q} \tag{14}$$

In order to show the proposed trust CTMC is an ergodic CTMC, the following condition must hold:

- *Necessary Condition* CTMC should be time homogeneous and irreducible: It is already shown that the proposed trust CTMC is time homogenous and irreducible.
- *Sufficient Condition* There exists a unique steady state vector of the CTMC: The proposed trust CTMC has strictly positive steady state or equilibrium probability vector n be gained from (14), when an additional normalization condition is imposed. Because beside the trivial solution $\pi_i = 0\ \forall i \in S$, any vector obtained from this also yields a solution. This can be expressed in vector matrix form we introduce the unit vector $\mathbf{1} = [1, 1, 1, \ldots, 1]^T$, so that

$$\pi.\mathbf{1} = \pi.\sum_{i\in S} \pi_i = 1 \tag{15}$$

satisfies the sufficient condition to prove the proposed trust CTMC is an ergodic CTMC. To analyze the steady trust state of each member node we assume that the change of trust state is a Markov chain model. Here, we precisely define different events for which trust state is changed. A good manner that is cooperating behavior increases the trust state and non-cooperating behavior decreases the trust state.

4.3 Trust state diagram

The trust state diagram is depicted in Fig. 4. Comparing the quantitative and analytical model of trust, it is evident that CH blacklists a node if its trust value is less than 0.25. The *blacklisted* state is represented in the Markov model as state 0. As we are concerned with selfishness mitigation, here packet dropping and reviving from *selfish* to *good* state have assigned higher priority. Increment and decrement of trust state due to different cooperative and uncooperative behaviors and weights can be varied depending on different application scenarios.

5 Analysis of the proposal

The attacks in MANET can roughly be classified into two major categories, namely passive attacks and active attacks. A passive attack obtains data, exchanged in the network without disrupting the operation of the communications or the network topology, while an active attack involves information interruption, modification, or fabrication,thereby disrupting the normal functionality of a MANET. As we are basically concentrating on the network layer, we try to categorize the network layer attacks and show our proposed protocol is able to find and prevent those attacks. Following are the major attacks that be seen in MANET and also we briefly discuss how those attacks can be identified and prevented by our proposed routing model.

- *Flooding attack* Any malicious node can broadcast fake RREQ for non-existence destination or collaborative with another malicious destination node. Other nodes in the network receive this message and rebroadcast it throughout the network. Therefore, it consume bandwidth and energy of the node for flooding the message in the network. This type of attack is known as flooding attack and our proposed scheme can prevent it in the following way: any node wants to communicate with another node, it has

to get *Permit-Route* from CH. If the source/destination nodes are non-malicious, non-selfish and destination node is exist in the network, CH broadcast the *RREQ* message along with its signature. If any node finds *RREQ* message without signature of the CH, it drops the message instead of further flooding or sending route reply. In addition, false injection by a node carefully measured in the trust calculation phase thus the chance of flooding attack eliminates further.

- *Selfishness* In our proposed scheme, if a node drops packet higher than a permissible threshold, its trust value is decreased. Also, calculating the reputation value of a node, any node is categorized as *good, selfish or malicious*. We categorically analyze the route discovery for all types of nodes:
  - Only *trusted and good* nodes are allowed to communicate between each other in a cluster or different clusters.
  - If the source/destination node is *malicious*, *Permit-Route* is not granted by CH.
  - If the source/destination node is *selfish*, our protocol forcefully send the node in *WAIT* state. The proposed scheme supports second chance mechanism to revoke its status from *selfish* to *good* by proper forwarding of packets.
  - If any intermediate node is *selfish* but *trusted*, route may be established through this node thus giving a chance to revive its status from *selfish* to *good*.

  Also, a rewarding scheme for every proper forwarding is developed to encourage the nodes to behave truthfully and to increase their trust and reputation rating by proper participation in the network functionalities. Any node tries to communicate with other node must have a minimum protocol defined amount of *VC*. Because both source and destination have to pay to intermediate nodes which are in the route. By every proper forwarding each intermediate node gets *VC* and cost of the packet is increased with per hop traveling. The total cost is equally debited from source and destination nodes. At the route discovery phase, CH checks whether the source-destination pair can pay the packet cost or not. If CH finds, they are lacking of *VC*, they hold in WAIT state until they can earn the required *VC*, which is used in future communication. Nodes are forcefully compelled to participate in routing otherwise they cannot earn *VC* for successful transmission of their own packets.
- *Collusion attack* In this attack, an attacker records packets at one location in the network and tunnels them into another location. Routing can be disrupted when routing control messages are tunneled. This tunnel between two colluding attackers is referred as a wormhole. Wormhole attacks are severe threats to MANET routing protocols. To revive from selfishness, two/three colluding nodes may create a wormhole. Just to advertise itself as *good*, a malicious node M can send only packets to another colluding node N to increase its trust value. Gradually it revives from *selfish* to Good. This type of attack can be detected and prevented using this model. If GN finds repeated communication within a short period of time, it generates a Alert to CH and consequently CH alerts the node and this malicious behavior can be prevented by dropping the *Permit-Route*.
- *Blackhole attack* The blackhole attack has two properties. First, the node exploits the routing protocol, such as AODV, to advertise itself as having a valid route to a destination node, even though the route is spurious, with the intention of intercepting packets. Second, the attacker consumes the intercepted packets without any forwarding. If GN finds any node does not forward any packet, the trust value and status is automatically decreased and this node will be blacklisted. So blackhole attack can be detected and prevented.
- *Grayhole attack* This is a more subtle form of blackhole attacks when an attacker selectively forwards packets. An attacker suppresses or modifies packet originating from some nodes, while leaving the data from the other nodes unaffected, which limits the suspicion of its wrongdoing. If any GN finds any node M does not forward all packets it receives, the status of the misbehaving node becomes selfish as the packet drop ratio of the said node reaches the allowable threshold. CH generates an Alert message to *M* and blocks until it revive from selfish to good. If *M* continues misbehaving it will be detected as Malicious and will be blacklisted. Similarly using GN, Byzantine attack, rushing attack and resource consumption can be mitigated with this trust model.
- *Cheating attack* Any malicious node may falsely report the trust manner about its neighbor node to CH, while CH is asking for recommendation trust. This type of attack is known as cheating attack and it can be detected and prevented by our trust model in the following way: as the resultant trust is calculated using weighted combination of trust opinions generated by different nodes along with CH. Here, highest priority is assigned to CH's own opinion. Therefore, cheating attack by any node has not much impact on evaluating trust of a node under consideration. If CH finds any node (which is already been detected as malicious) falsely reports about other nodes, CH does not consider the opinion of the said node at the time of trust evaluation. Even if CH finds any node (which is still not detected as malicious) falsely reports about other nodes, CH generates WARN message to that misbehaved node. If the misbehaved node continues such bad behavior, it is isolated from network.
- *ON/OFF attack* In this attack a malicious node behaves good and bad alternatively and CH cannot determine the

trust state of the said node correctly. In our proposal, it can be detected and prevented in the following way: each mobile node in a cluster is monitored by more than one GN. If any GN fails to monitor the random good and bad behavior of any malicious node, there is a very less chance that other GNs fail to monitor it.

– *Sybil attack* If any malicious node blacklisted from a cluster, the node may try to join another cluster as a new node by hiding its identity. This attack is known as Sybil attack. When a node is blacklisted by CH_1, it informs neighbor CHs about the said node by sending its public key. Therefore, that node cannot join to any cluster with the same public key.

6 Simulation

Our proposed STACRP has been implemented on top of AODV in NS-2 (version-2.34) simulator [29] on Fedora Core 9 platform. We evaluate the performance of the STACRP and AODV with various percentage of selfish nodes. *Packet delivery ratio*, *throughput*, *routing overhead* and *routing latency* are chosen as the performance comparison metrics and results are demonstrated here in this section. *Throughput* is defined as the number of bits transmitted per unit time and *packet delivery ratio* is the ratio between the number of data packets (i.e. Constant Bit Rate) sent and the number of received packets by the CBR sink at destination. *Routing overhead* is refer to average number of routing message exchange per node per CBR data connection to form a valid route and *routing latency* is refer to total time needed to form that route between source and destination nodes.

6.1 Simulation parameters

In our simulation, the radio transmission range of nodes was 250 m and the underlying MAC protocol defined by IEEE 802.11. TCP was used as the transport layer protocol with Constant Bit Rate (CBR) traffic generator of packet size 512 bytes. A node joins the MANET every 5 s using IDDIP scheme [30] and forms the cluster [28] in case our proposed STACRP scheme. The random way point mobility model was adopted, wherein node speeds were randomly distributed between 0 m/s to 10 m/s and on reaching the destination the pause time of a node was set to 5 s. The propagation model used was two ray ground and total simulation time was set to 100 s for each set of simulations. The simulation was done for 10 number of mobile nodes with 330 m $\times$ 330 m network area. Simulation included four CBR data connection, each of which generated four packets per second. The selfish nodes were made to drop the messages that they were to forward. Summary of the simulation parameters are presented in Table 2.

Table 2 Simulation parameters & environment

Simulation parameter	Assigned value
Application agent	CBR
Packet size	512 bytes
Packet rate	4
Transport agent	TCP
Routing protocol	Proposed STACRP, AODV
Total number of nodes	10
Number of selfish nodes	1, 2, 3, 4
Addressing scheme	IDDIP
Mobility	0–10 m/s
Pause time	5 s
Simulation time	100 s
Mobility model	Random way-point

6.2 Simulation results & observations

In the following subsection, we demonstrate the results and observations of the proposed STACRP along with AODV in terms *Packet delivery ratio*, *throughput*, *routing overhead* and *routing latency*.

6.2.1 Packet delivery ratio & throughput

As number of selfish nodes increase in the network, number of active good node decreases. Thus availability of nodes in route establishment decreases. In the worst case, no path may be established between source and destination if the intermediate nodes are selfish and denied to forward others' packet. It may also happen that selfish nodes are there on a route and drops the data packet. As a result, acknowledgments from the destination are missing, the source node of a TCP session may slow down or even stop sending packets. Both the protocol gives more than 85% packet delivery ratio when there is no selfish node. In case of AODV, as number of selfish nodes increases, packet delivery ratio decreases significantly around 65% when 40% of the nodes are selfish. This is because no selfishness mitigation policy is incorporated with original AODV. On the other hand, in proposed STACRP, cooperation is enforced to mitigate selfishness. From Fig. 5, it can be seen that though the number of selfish nodes increases, the packet delivery ratio does not considerably lessen. Packet delivery ratio is near about 80% in the presence of 40% selfish nodes in the cluster. Because nodes are forcefully compelled to take part into network functionality otherwise they would be detected as malicious and eventually isolated from the network. Similarly throughput is substantially reduced with increment of selfish nodes in the network for AODV. From Fig. 6, we can observe that both protocols give good throughput when there is no selfish node. However, as number of selfish node increases,

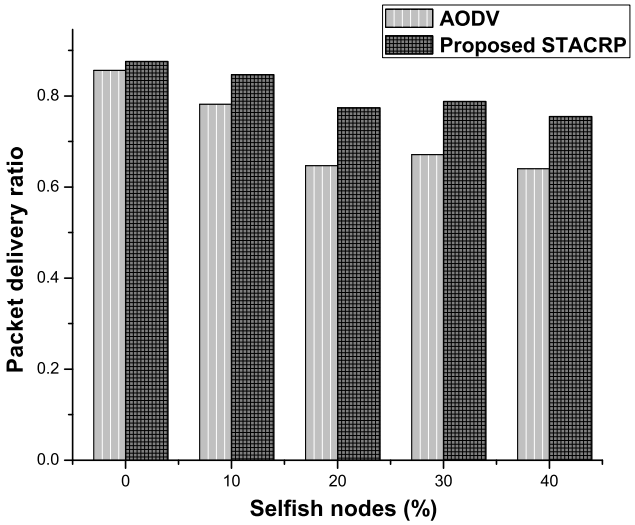

Fig. 5 Packet delivery ratio vs Selfish nodes

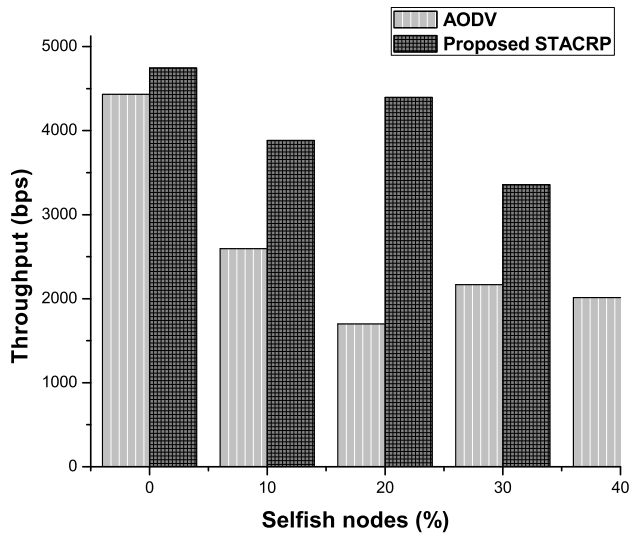

Fig. 6 Throughput vs Selfish nodes

the performance of AODV degrades significantly. On the other hand, the proposed STACRP gives steady throughput in presence of selfish node. In presence of 40% selfish nodes STARCP gives throughput around 4000 bps where as in AODV it becomes almost half (around 2000 bps). Thus it can be easily perceived that the proposed scheme effectively mitigate selfishness and enforce cooperation between nodes and increase nodes availability.

6.2.2 Routing overhead & latency

Figure 7 shows the average routing overhead per connection of the proposed STACRP along with AODV protocol. We varied the percentage of selfish node (from 0 to 40%) and study the following: It can be seen that the average routing

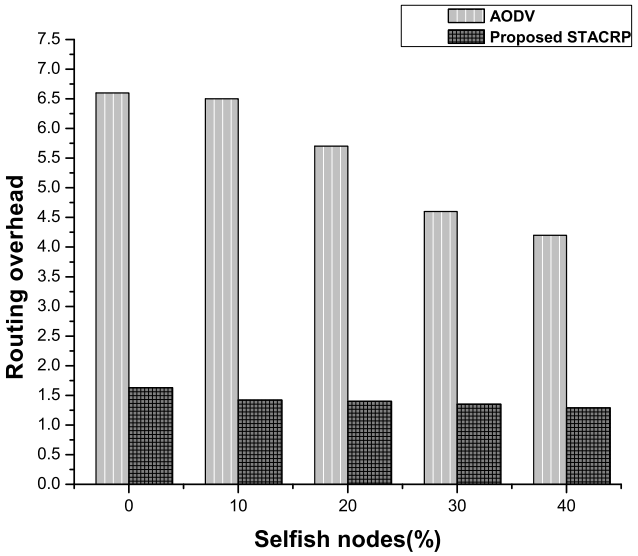

Fig. 7 Routing overhead Vs Selfish nodes

overhead decreases as the number of selfish nodes increases because selfish nodes drop RREQ. It varies a little for the proposed scheme due to enforces cooperation and nodes are compelled to forward others' packets. Compared with the AODV scheme, the propose STACRP has much lower average routing overhead. This is because in AODV, RREQ originated from source is flooded throughout the network until destination or one hop before to destination is reached. So, number of message exchange in AODV increases as nodes broadcast the routing packets. As the number of selfish node increases, the overhead is decreased because selfish nodes drop packets. On the other hand, the proposed STACRP has an advantage over AODV that it supports 1-hop clustering. In STACRP, source node sends the RREQ with the is_neighbor flag (1, if the destination node is 1-hop; 0, otherwise) to the CH. If the is_neighbor flag is 1, CH unicasts the RREQ to destination and destination sends the RREP directly. If not, CH broadcasts the RREQ, all nodes including destination receive the RREQ. On reception of the RREQ, a node checks neighbor list and there may arise two possible cases: Either source and destination is neighbor of any node, RREP is unicasted to source. Thus, re-broadcasting or network-wide flooding of RREQ can be avoided in average cases. In another case, route is established by rebroadcasting RREQ until destination is reached. Though the number of selfish node increases in the network, it does not affect the routing overhead. This is due to fact that cooperation is enforced in the proposed protocol and RREQ coming from selfish node is not entertained at all.

From Fig. 8, it can be seen that as the number of selfish node increases routing latency also decreases for AODV. Because as selfish nodes increases the packet drop increases and less number of nodes take part into routing. So, the num-

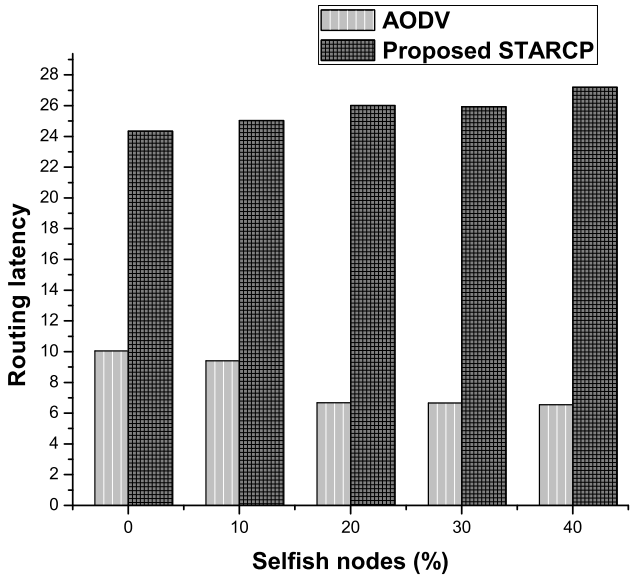

Fig. 8 Routing latency vs Selfish nodes

ber of RREQ broadcasting decreases thus latency also decreases. In the proposed STACRP, routing latency is significantly higher than that of AODV. As we consider auction mechanism to determine the cost of the packet at each node timers are kept to buffer the messages to find the optimum bidder. So, longer time is needed to establish a route between source and destination with optimum bidding.

7 Conclusions

In this paper, we propose a secure trusted auction oriented clustering and routing protocol (STACRP) that can effectively detect selfish nodes and enforce cooperation between nodes. The proposed scheme uses both credit and reputation based scheme to mitigate selfishness. *Virtual currency VC* is rewarded to the intermediate node for forwarding others' packets. This *VC* can be used to forward its own packet in future. Any node is compelled to others packet because firstly, forwarding others' packet is the only way to earn *VC* and without enough *VC* its own packet would not be forwarded; secondly, if a node drops packet its trust value decreases and if the value becomes less than the predefined threshold, the node would be detected as malicious and get isolated from the network; thirdly once a node becomes *trusted* and *good*, it has to pay very little *VC* to forward its own packet. So, to forward its own packet in future and to maintain its own membership in the cluster as well as in the network a node has to forward others' packet. Thus cooperation is assured by the proposed scheme. A second chance mechanism is incorporated in the proposal for selfish node to revoke its status from *warned* to *good* by proper forwarding of others' packets. The trust model is analyzed and compared with Markov chain and the proposed model is proved as an ergodic continuous time Markov chain. Attack model and security analysis of the protocol shows that this model gives secure intra cluster routing backbone for MANET. Simulation results show that our scheme achieves better packet delivery ratio and throughput with less routing overhead than AODV.

Appendix

A survey on existing enforcing cooperation and mitigating selfishness schemes between nodes along with our proposed STACRP scheme for MANET are given in Table 3. Description of different cooperation enforcement schemes are adapted from [31] to compare our proposal. The meaning of the field entry in Table 3 as follows:

– *Reputation/Payment: Reputation* means self or recommendation evidences from trusted neighbors. *Payment* is rewarding scheme for good behavior.

Table 3 Comparison of existing cooperation enforcement schemes

Protocol [Ref. no]	Payment/ reputation	Misbehavior detection	Type of observation	Defense for collusion	Cryptography authentication	Promiscuous observation	Trust considered
CONFIDANT [10]	Reputation	General	Global	✓	X	✓	Pre-existed trust assumed
CORE [11]	Reputation	Selfishness	Global	X	X	✓	X
OCEAN [12]	Both	Selfishness	Global	X	X	✓	Pre-existed trust assumed
SPRITE [2]	Payment	General	–	✓	✓	X	X
TOKEN [3]	Payment	General	–	✓	✓	✓	A-priory trust assumed
VCG [5]	Payment	Selfishness	–	X	X	X	X
Liu et al. [32]	Reputation	General	Context	✓	X	–	X
STACRP-proposed scheme	Both	General	Global	✓	✓	✓	Trust evaluated online in dynamic distributed manner

– *General/Selfishness: General* means selfishness as well as other misbehavior of any node. *Selfishness* is described by typical behavior of a node to save battery for its own communication and jeopardizing network functionality by not participating in routing.
– *Global/Context: Global* observation means a node monitors or traces network packets as a whole and *context* refers if the monitoring node traces network traffic only on some specific contexts.

References

1. Buttyan, L., Hubaux, J.P.: Nuglets: a virtual currency to stimulate cooperation in self-organized mobile ad hoc networks, 2001
2. Zhong, S., Chen, J., Yang, Y.R.: Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks. In: Proceedings of IEEE INFOCOM 2003, pages, pp. 1987–1997, March–April (2003)
3. Yang, H., Meng, X., Lu, S.: Self-organized network-layer security in mobile ad hoc networks. In: Proceedings of ACM WiSe02, September (2002)
4. Perkins, C., Royer, E.B., Das, S.: Ad hoc on demand distance vector (aodv) routing. IETF RFC 3561, July 2003
5. Anderegg, L., Eidenbenz, S.: Ad hoc-vcg: A truthful and cost efficient routing protocol for mobile ad hoc networks with selfish agents. In: Proceedings of MobiCom 2003, pp. 245–259, September (2003)
6. Hu, Y.-C., Perrig, A., Johnson, D.: The dynamic source routing protocol for mobile ad hoc networks (dsr). draft-ietf-manet-dsr-10.txt, July 2004
7. Buchegger, S., Boudec, J.-Y.L.: Performance analysis of the confidant protocol (cooperation of nodes: Fairness in dynamic ad-hoc networks). In: Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), 9–11 June (2002)
8. Hu, J.: Lars a locally aware reputation system for mobile ad hoc networks. In: Proceedings of 44th Annual Southeast Regional Conference, pp. 119–123 (2006)
9. Marti, S., Giuli, T.J., Lai, K., Baker, M.: Mitigating routing misbehavior in mobile ad hoc networks. In: Proceedings of 6th Annual International Conference on Mobile Computing and Networking, MobiCom and 2000, pp. 255–265, August (2000)
10. Buchegger, S., Boudec, J.: Performance analysis of the confidant protocol cooperation of nodes fairness in dynamic ad-hoc networks. In: Proceedings of International Symposium on Mobile Ad hoc Networking and Computing MobiHoc and 2002, pp. 226–236, June (2002)
11. Michiardi, P., Molva, R.: Core a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In: Proceedings of Communications and Multimedia Security ConferenceCMS and 2002, September (2002)
12. Bansal, S., Baker, M.: Observation-based cooperation enforcement in ad-hoc networks (2003)
13. Milan, F., Jaramillo, J.J., Srikant, R.: Achieving cooperation in multihop wireless networks of selfish nodes. In: Proceedings of Workshop on Game Theory for Networks GameNets 2006, October (2006)
14. Adams, W.J., Davis, N.J.: Toward a decentralized trust-based access control system for dynamic collaboration. In: Proceedings of 6th Annual IEEE SMC Information Assurance Workshop IAW'05, pp. 317–324, June (2005)
15. Boukerche, A., Ren, Y.: A security management scheme using a novel computational reputation model for wireless and mobile ad hoc networks. In: Proceedings of Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems, pp. 88–95 (2008)
16. Li, R., et al.: An objective trust management framework for mobile ad hoc networks. In: Proceedings of IEEE 65th Vehicular Technology Conf. VTC'07, pp. 56–60, April (2007)
17. Theodorakopoulos, G., Baras, J.S.: Trust evaluation in ad-hoc networks. In: Proceedings of Workshop on Wireless Security, pp. 1–10 (2004)
18. Jiang, T., Baras, J.S.: Ant-based adaptive trust evidence distribution in manet. In: Proceedings of 2nd Int'l Conf. on Mobile Distributed Computing Systems Workshops and MDC, pp. 588–593, March (2004)
19. Yan, Z., Zhang, P., Virtanen, T.: Trust evaluation based security solution in ad hoc networks. In: Proceedings of Seventh Nordic Workshop on Secure IT Systems (2003)
20. Pirzada, A.A., McDonald, C.: Establishing trust in pure ad-hoc networks. In: Proceedings of 27th conference on Australasian computer science CRPIT'04 and Australian Computer Society and Inc (2004)
21. Ghosh, T., Pissinou, N., Makki, K.: Collaborative trust-based routing in multi-hop ad hoc networks. In: Proceedings of Proc. 3rd Int'l IFIP-TC06 Networking Conf. Lecture Notes in Computer Science, pp. 1446–1451, 9–14 May (2004)
22. Ghosh, T., Pissinou, N., Makki, K.: Towards designing a trust routing solution in mobile ad hoc networks. Mob. Netw. Appl. **10**, 985–995 (2005)
23. Balakrishnnan, V., Varadharajan, V., Tupakula, U.K., Lucs, P.: Trust and recommendations in mobile ad hoc networks. In: Proceedings of 10th IEEE International Conference on Networking and Services, pp. 64–69, 19–25 June (2007)
24. Wang, X., Liu, L., Su, J.: Rlm: A general model for trust representation and aggregation. IEEE Trans. Serv. Comput. **99** (2010)
25. Virendra, M., Jadliwala, M., Chandrasekaran, M., Upadhyaya, S.: Quantifying trust in mobile ad-hoc networks. In: Proceedings of IEEE International Conference on Integration of Knowledge Intensive Multi-Agent Systems KIMAS'05, pp. 65–71 (2005)
26. Chang, B.-J., Kuo, S.-L.: Markov chain trust model for trust-value analysis and key management in distributed multicast manets. IEEE Trans. Veh. Technol. **58**, 1846–1863 (2009)
27. Cho, J., Swami, A., Chen, I.: A survey on trust management for mobile ad hoc networks. IEEE Commun. Surv. Tutor. **99**, 1–22 (2010)
28. Chatterjee, P., Sengupta, I., Ghosh, S.: A trust based auction oriented routing model for ad hoc networks. In: Proceedings of IEEE TrustCom and 2010 (2010)
29. Fall, K., Varadhan, K.: ns manual. isi.edu/nsnam/ns/doc
30. Ghosh, U., Datta, R.: Iddip: An id based secure dynamic ip configuration scheme for mobile ad hoc networks. In: Proceedings of the First International Conference on Network and Service Security, 24–26 June (2009)
31. Marias, G., Georgiadis, P., Flitzanis, D., Mandalas, K.: Cooperation enforcement schemes for manets: A survey. Wirel. Commun. Mob. Comput. **6**, 319–332 (2006)
32. Liu, J., Issarny, V.: Enhanced reputation mechanism for mobile ad hoc networks. In: Proceedings of 2nd International Conference on Trust Management, March (2004)

Pushpita Chatterjee is currently a Ph.D. scholar at School of IT in the Indian Institute of Technology (IIT) Kharagpur, India. She did her M.Tech. and M.Sc. degrees in Computer Science and Engineering (CSE) from the University of Calcutta. Her current research area is trust based wireless network security. She has good number of research publications in reputed conferences and journals. Her research interests include mobile computing, distributed computing and wireless ad hoc and sensor networks. She is a student member of IEEE.

Indranil Sengupta has obtained his B.Tech., M.Tech. and Ph.D. degrees in Computer Science and Engineering from the University of Calcutta in 1983, 1985, and 1990, respectively. He joined Indian Institute of Technology, Kharagpur, as a faculty member in 1988, in the Department of Computer Science and Engineering, where he is presently a Professor at the Department. Dr. Sengupta is former head of this department and also was heading the School of IT of the Institute. He has over 24 years of teaching and research experience, and over 100 publications to his credit in international journals and conferences. A Centre of Excellence in Information Assurance has been set up at IIT Kharagpur under his leadership, where a number of security related projects are presently being executed. His research interests include cryptography and network security, side-channel attacks on cryptosystems, VLSI design and testing, and mobile computing. He is a member of IEEE.

S.K. Ghosh did his M.Tech. and Ph.D. in Computer Science & Engineering from the Indian Institute of Technology (IIT) Kharagpur, India. He is currently an Associate Professor at the School of Information Technology, IIT Kharagpur. Before joining IIT Kharagpur, Dr. Ghosh worked for Indian Space Research Organization in the area of Satellite Remote Sensing and GIS. His research interests include Network Security and Spatial Web Services. He has over 70 research papers in reputed journals and conference proceedings. He is a member of IEEE.