Chapter 2

# A Composite Trust Model for Secure Routing in Mobile Ad-Hoc Networks

Rutvij H. Jhaveri, Narendra M. Patel and Devesh C. Jinwala

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/66519

**Abstract**

It is imperative to address the issue of secure routing in mobile ad-hoc networks (MANETs) where the nodes seek for cooperative and trusted behaviour from the peer nodes in the absence of well-established infrastructure and centralized authority. Due to the inherent absence of security considerations in the traditional ad-hoc routing protocols, providing security and reliability in the routing of data packets is a major challenge. This work addresses this issue by proposing a composite trust metric based on the concept of social trust and quality-of-service (QoS) trust. Extended from the ad-hoc on-demand distance vector (AODV) routing protocol, we propose an enhanced trust-based model integrated with an attack-pattern discovery mechanism, which attempts to mitigate the adversaries craving to carry out distinct types of packet-forwarding misbehaviours. We present the detailed mode of operations of three distinct adversary models against which the proposed scheme is evaluated. Simulation results under different network conditions depict that the combination of social and QoS trust components provides significant improvement in packet delivery ratio, routing overhead, and energy consumption compared to an existing trust-based scheme.

**Keywords:** packet-forwarding misbehaviour, secure routing, composite trust model, attack pattern discovery, mobile ad-hoc networks

## 1. Introduction

A mobile ad-hoc network (MANET) is an autonomous system of wireless mobile nodes that dynamically form a network in order to exchange information in the absence of centralized authority and fixed infrastructure. Mobile nodes communicate with each other in a multi-hop way to carry out data transmission due to limited communication range and resource

constraints of the nodes. In the absence of router, each node operates as a host as well as a wireless router to forward packets for other nodes that may be outside its communication range [1]. The network functions well if all nodes operate in an altruistic manner. Due to the openness in network topology, distributed nature and lack of central authority, MANETs are particularly vulnerable to different types of routing attacks launched by internal nodes [2]. As a result, routing in such dynamic networks faces inherent challenges as compared to the traditional wireless networks. The traditional routing protocols proposed for ad-hoc networks are inefficient in dealing with different routing attacks.

The security schemes based on traditional cryptographic systems are typically used to resist external attacks. However, they prove to be inefficient in resisting the attacks launched by internal malevolent nodes. Such malicious nodes may seriously influence the security of the network by performing distinct types of packet-forwarding misbehaviours. In such a hostile environment, introducing the concept of 'trust' would provide prediction about the behaviour of neighbour nodes [2]. The notion of trust would prove to be useful for dynamic environments where the nodes need to depend on each other to achieve their goals [3]. Recently, trust management schemes have been considered as a viable security solution to improve the routing decisions in MANETs by detecting and isolating distrusted nodes [4].

In our previous work [5], we devised a trusted routing scheme with pattern discovery (TRS-PD) that integrates a trust model (based on QoS trust components) with an attack-pattern discovery mechanism in order to detect the malicious nodes earlier than a solitary trust model. TRS-PD estimates the distrust degree of neighbour nodes using direct trust computation. On the top of this, the attack-pattern discovery mechanism is introduced, which predicts suspicious activities of the neighbour nodes by promiscuously monitoring and recording specific fields of the control packets which are transmitted by the neighbour nodes. This gives an idea about the neighbour nodes, which might be following certain attack patterns. In addition, the scheme carries out indirect computation using recommendations by the trusted neighbours in order to enhance the trust establishment process. In this chapter, we propose enhanced TRS-PD (ETRS-PD), which uses a composite trust model that combines social trust component along with QoS trust components. ETRS-PD attempts to improve the packet delivery ratio against the adversary models discussed in Ref. [5] by enhancing the routing process. The performance of ETRS-PD is compared with TRS-PD against these adversary models under different network conditions.

The main technical contributions of this work are as follows: (1) An enhanced trust model is proposed for AODV protocol to evaluate neighbours' distrust value using composite trust metric. (2) Simulations carried out to compare the performance of ETRS-PD with TRS-PD prove that the performance of MANETs employing ETRS-PD is superior to that of MANETs employing TRS-PD against distinct types of adversaries.

The rest of the chapter is organized as follows. Section 2 discusses relevant related work. In Section 3, the proposed trust model is discussed. The enhanced trust-based on-demand routing scheme incorporated into AODV protocol is discussed in Section 4. Section 5 presents operations performed by various adversary models. The simulation results depicting the performance of ETRS-PD are presented in Section 6. Finally, Section 7 concludes the chapter.

http://dx.doi.org/10.5772/66519

## 2. Related work

A substantial amount of research work has been carried out in the last few years to address the security requirements of routing protocols by means of trust management.

A trust-based source routing (TSR) scheme devised by Xia *et al.* [6] attempts to discover a shortest secure route for data transmission in MANETs. Neighbour nodes are evaluated based on the historical trust values using correct packet-forwarding ratios. In addition, fuzzy logic is used to estimate a node's current trust based on its capability and historical trust value. This estimated value is used to predict the misbehaving nodes in the neighbourhood. A trusted route is selected for data transmission by avoiding such untrustworthy nodes. Experimental results show the effectiveness of TSR against blackhole, grayhole and modification attacks. However, the scheme incurs high computational overhead in calculation of route trust after arrival of every data packet at the destination. Furthermore, the scheme requires buffering of the packets in a circular queue, which incurs significant overhead in searching the match for the packets in the buffer. Gharehkoolchian *et al.* [7] proposed a novel trust model, which uses different *trust levels (TL)* for nodes and imposes the limitations based on the trust level in order to mitigate the malicious nodes. When a node enters the network, it is assigned $TL = 1$. It gains higher reputation if it acts normally by forwarding packets and thereby, it is assigned $TL = 2$. In case of malicious behaviour, it is assigned $TL = 0$. If the malicious behaviour of the node is observed for three times, it is assigned $TL = -1$ and the node is permanently blocked. During the route discovery process, when a node receives a route reply from its neighbour node, it verifies its $TL$ value. If the node is a non-malicious node $(TL = 2)$, the route reply is forwarded. Otherwise, a test route request is sent to the suspicious node $(TL = 1)$ after the received route reply. If an abnormal reply is received from the suspicious node in response to the test route request, the route reply is discarded after assigning $TL = 0$ to the node and the node is blocked for a specific time. Thus, the scheme attempts to isolate the malicious nodes during route discovery process. However, it does not have any reactive mechanism to cope up with sudden drops in packets during data transmission phase; instead, it just detects the adversary but attempts to isolate it during the next route discovery process. Airehrour *et al.* [8] proposed *GradeTrust* protocol to isolate blackhole adversaries by selecting a secure path, in addition to elimination of excessive routing computations and minimization of communication overhead. It classifies the nodes into three sets in order of the trust levels: *Trusted Friends*, *Friends* and *Possible Friends*. Trust level is assigned by monitoring neighbours' request packet-forwarding ratio. A source node selects the next hop from its Trusted Friends, and the process continues until the packet reaches the destination. In the case of unavailability of a Trusted Friend, a Friend is selected. A compromised node is dissociated swiftly from other trusted nodes, and it is pushed down to the lower trust level. However, the scheme does not consider the forwarding ratio of data packets in calculation of the trust level, which makes it susceptible to packet dropping adversaries during data transmission phase. In addition, simulation results showing comparison of the proposed protocol with the traditional protocols are not promising. Patel *et al.* [9] proposed a trust model for AODV-based MANETs, which attempts to increase network lifetime by uniform consumption of energy. A trust value is computed based on dropping ratios and delays of control and data packets as well as residual node

energy. The scheme attempts to discover a trusted route during the route reply propagation towards the source node on the reverse path. All the intermediate nodes receiving the route reply packet update the path trust value in the packet using the available trust values of neighbours. If a node receives multiple route reply packets, it compares the trust of the newly received path with that of the current path and stores the path with the maximum trust value. However, the scheme does not have any reactive mechanism to fight against packet-dropping adversaries during data transmission phase. After identifying an adversary, it waits for the next route discovery process to isolate it. Chiejina *et al.* [10] proposed a solution to evaluate the trust of a node in the network, which ensures that nodes expending their energy in forwarding data and control packets for other nodes are allowed to carry on their activities while the malevolent nodes are isolated from the network. Trust values are computed by direct observations, which are aggregated at different time intervals to provide a historical reputation of the node. The total reputation value of a node is mapped with a grading criterion to decide the status of a node. Nodes with lower reputation value than the set threshold value are blacklisted and denied the network resources. Routes containing blacklisted nodes are discarded, and alternative routes are discovered. The solution attempts to mitigate selfish and deceitful nodes from the network with scarce resources. However, whenever the source sends a packet towards the destination, the solution generates additional overhead as *path administrator* has to check that the packet has not been sent via a path containing a blacklisted node. Mylsamy *et al.* [11] proposed a *preference-based protocol for trust and head selection (2PTH),* which takes four parameters to calculate a trust value: packet delivery ratio, packet misrouting ratio, packet alteration ratio, and packet injection ratio. Depending on the affected security parameters, weighing coefficients' values are determined. Trust values are classified into three different categories: high, medium and low. If trust value of a node goes below its relative threshold, it is not allowed to participate as a cluster member. A cluster-based routing mechanism is used which discovers a stable cluster head based on external factors such mobility, connectivity and distance as well as internal factors such as residual battery power, processing power and memory. When a cluster head of the cluster of the destination node receives a route request packet during the route discovery process, it verifies the trustworthiness of the node in order to establish a secure route. Simulation results show promising performance of the protocol as compared to some existing protocols. However, the protocol does not have any reactive mechanism for identifying packet-dropping adversaries during data transmission phase. Moreover, *weight assignment* and *cluster head election* consume a significant amount of computational resources. Indirani *et al.* [12] presented a *swarm-based distributed intrusion detection system (SDIDS)* with the objective to remove the complexity in the design of an IDS caused by the inherent MANET characteristics. Active nodes in a route are selected by *ant colony optimization (ACO)* technique based on a node's packet-forwarding activities, residual bandwidth, residual energy and connectivity. A *forward ant* reaches to every node in order to compute and update the pheromone value using the aforementioned parameters. When it reaches the destination, the information collected by the forward ant about all the hops is transferred to the *backward ant*. The backward ant then traverses on the reverse path and reaches to the source in order to deliver the status of all nodes. A routing decision is then made by selecting the optimal route to the destination. However, the scheme incurs high computational overhead in calculation of route trust. In addition, establishment of a trusted

http://dx.doi.org/10.5772/66519

route should not be the sole responsibility of the source node. Xia *et al.* [13] proposed a *light-weight trust-enhanced routing protocol (TeAOMDV),* which attempts to provide an optimal two-way trusted route without containing the malicious entities. Its trust framework uses passive and local monitoring information to evaluate the trust values of neighbours. It considers activity, stability and historical trust record of a node in evaluation of a node trust. Moreover, the trust value is modified by collecting the recommendations from the trusted neighbours. It uses *hop count*, *forward path trust* and *reverse path trust* as the metrics to compose a three-dimensional evaluation vector for taking routing decisions. The authors extend their work by proposing an improved *SCGM(1,1)-Markov chain prediction method* based on the *system cloud grey model* and *Markov stochastic chain theory* to forecast trust level of a node for future routing decisions. However, it holds similar drawbacks as the scheme proposed in [12] due to the consideration of route trust. Azer *et al.* [14] proposed a new reputation system for ad hoc networks, called *misbehaviour detection and control (MDAC),* which encourages the nodes to act in a trustworthy manner. It obtains first hand and second hand information about neighbouring nodes. Trust is evaluated based on number of incoming packets and total consumed time to deliver packets. The *MDAC modeller module* combines all collected information about a node into a meaningful reputation value. Based on the reputation values, nodes in the network are guided to take necessary actions such as trust/don't trust, cooperate/don't cooperate and forward/don't forward. A node is considered eligible for service only after verifying its reputation value. The mechanism shows better performance compared to an existing scheme in terms of throughput and delay. However, the mechanism does not consider control packets in the calculation of the reputation value which delays the detection of sequence number attacks. In addition, it adds significant computational overhead in making reputation decisions about neighbouring nodes. Rajkumar *et al.* [15] proposed a trust-based light-weight authentication routing protocol which adopts multipath route discovery technique to mitigate adversaries. A route is rated based on packet success rate after route reply is forwarded to the source node. An optimal path for data transmission is chosen based on its rating, and the next optimal path is stored as an alternative arrangement. The protocol calculates a trust value using *EigenTrust* algorithm, which is based on direct and indirect observations of neighbour nodes. A resolver is engaged for computing a global trust value of the node, which also executes trust noise cancellation mechanism. If the trust value of a node goes below the threshold value, it is authenticated using the *Shamir's secret sharing* technique. If a node is found to be malicious, all routes going through the node are discarded and the alternate optimal path is selected. However, cryptographic approaches add considerable amount of communication and memory overhead along with key distribution issues. In addition, the scheme involves high computational overhead in the estimation of packet success rate and calculation of the global trust value.

## 3. Trust model

As a part of the literature survey, we discover that a composite trust metric based on social and QoS trust components may successfully perform tasks to meet both performance and trust requirements [16, 17]. We have noticed some work in the literature moving in this direction.

Cho *et al.* [17] considered honesty and intimacy, while Kohlas *et al.* [18] considered honesty, competency, reliability and maliciousness as social trust components to define trust relationships. In addition, we observe that *energy consumption* is an important QoS trust component for improving the network performance [17, 19]. Taking these notes into consideration, we devise an enhanced trust-based scheme, *ETRS-PD*.

ETRS-PD considers *ditch ratio* as a social trust component in estimation of distrust degree of the neighbours. This social trust component is utilized to know the magnitude of misbehaviour carried out by a node while residing in monitoring node's neighbourhood. In addition, *energy consumption* is considered as an additional QoS component along with *packet drop ratio*. Thus, a composite trust metric is constructed by including social trust along with QoS trust. Furthermore, the routing process of TRS-PD is modified to enhance the routing decisions. As aforementioned, ETRS-PD attempts to improve the packet delivery ratio against the adversary models discussed in our previous work [5].

In our trust model, we compute historical trust on a constant basis after a specific time interval called trust update interval. Overall, our trust model performs trust derivation and trust computation along with discovery of attack patterns. We modify the trust model of TRS-PD to perform trust derivation and trust computation in a different way.

### 3.1. Basic assumptions

Our trust-based scheme makes the following assumptions: (i) all the mobile nodes have identical physical characteristics; (ii) the wireless links in the network are bidirectional; (iii) all the nodes operate in promiscuous mode in order to observe the neighbour nodes and (iv) the source and the destination are benevolent nodes. The above assumptions are fulfilled by wireless MAC layer protocols.

### 3.2. Trust derivation

Our proposed trust model uses direct observations to derive distrust values of neighbour nodes by observing packet dropping ratios, energy consumption and ditch ratio of neighbour nodes. In addition to this, each node employs an attack pattern discovery mechanism, which detects malicious patterns generated by neighbour nodes in the transmitted control packets. We also consider recommendations of trusted neighbours for improving the routing decisions.

### 3.3. Trust computation

In a routing process, neighbour node's distrust is evaluated by the sender by observing activities carried out by that neighbour. To be specific, a node $n_i$ will increase the distrust score of its neighbour $n_j$ if the $n_j$ does not forward the packet sent by $n_i$ [5].

**Definition 1.** *Control dropping ratio (CDR)*: It is the ratio of the number of control packets dropped to the number of control packets which are supposed to be forwarded. At time $t$, *CDR(t)* is computed as follows:

http://dx.doi.org/10.5772/66519

$$CDR(t) = \frac{NC_d(t)}{NC_a(t)} \quad (1)$$

where *NCd(t)* signifies the cumulative count of dropped control packets, and *NCa(t)* represents the total number of sent control packets from time *0* to *t*.

**Definition 2.** *Data dropping ratio (CDR)*: It is the ratio of the number of data packets dropped to the number of data packets, which are supposed to be forwarded. At time *t*, *CDR(t)* is computed as follows:

$$DDR(t) = \frac{ND_d(t)}{ND_a(t)} \quad (2)$$

where *NDd(t)* signifies the cumulative count of dropped control packets, and *NDa(t)* represents the total number of sent control packets from time *0* to *t*.

**Definition 3.** *Energy consumption (EC)*: It is the ratio of the energy consumed by a node to the initial energy of that node. When a node possesses limited residual energy, it may not hold the capabilities to forward the packets of other nodes. At time *t*, *EC(t)* is computed as follows:

$$EC(t) = \frac{EI - ER(t)}{EI} \quad (3)$$

where *EI* signifies the initial energy, and *ER(t)* signifies the residual energy of the node at time *t*.

**Definition 4.** *Ditch ratio (DTR)*: It is the ratio of the number of times a neighbour node is found to be distrusted while receiving its *HELLO* packets to the total number of *HELLO* packets received from that node. At time *t*, *DTR(t)* is computed as follows:

$$DTR(t) = \frac{NH_d(t)}{NH_a(t)} \quad (4)$$

where *NHd(t)* signifies the number of times a distrusted neighbor node has ditched the monitoring node while sending *HELLO* packets, and *NHa(t)* signifies the total number of *HELLO* packets received from that neighbour node.

The obtained distrust value of a node *nj* by a monitoring node *ni* is the measure of packet dropping activities, energy drain rate and magnitude of misbehaviour. The distrust value of node *nj* evaluated by node *ni*, denoted as *DTVij*, is calculated by the following formula:

$$DTV_{ij}(t) = w1 \times CDR_{ij}(t) + w2 \times DDR_{ij}(t) + w3 \times EC_{ij}(t) + w4 \times DTR_{ij}(t) \quad (5)$$

where *w1, w2, w3* and *w4* (*w1, w2, w3, w4* ≥ 0 and *w1 + w2 + w3 + w4 = 1*) are the weights assigned to *CDR*, *DDR, EC* and *DTR*, respectively.

In our trust model, distrust values are restricted in the range from 0 to 1 (i.e., $0 \leq DTVij \leq 1$). The distrust value 0 indicates complete trust, whereas the distrust value 1 signifies complete distrust. We set the initial value of distrust to 0 as we assume all the nodes to be benevolent initially. Meanwhile, the distrust value constantly varies with the time as per the behaviour of neighbour nodes. We use a distrust threshold $\eta$ to differentiate the malicious nodes from benign nodes.

As discussed in Ref. [5], we incorporate an attack pattern discovery mechanism on the top of the trust model, which employs the model of *method of common differences (MCD).* Thus, the

pattern discovery mechanism attempts to identify the adversaries following attack patterns prior to conducting misbehaviours; on the other hand, the trust model detects other packet-dropping adversaries during the trust update procedure.

## 4. Enhanced trust-based on-demand routing

While any reactive routing protocol can be extended to incorporate ETRS-PD, we extend ad-hoc on-demand distance vector (AODV) protocol for this purpose. In addition to the modifications described in [5], we further modify the functionality of AODV in order to improve the routing decisions. The neighbour table is modified by appending the following fields: (i) *Energy consumption,* (ii) *Ditch count:* The number of times a neighbour node is found to be distrusted while receiving its *HELLO* packets, (iii) *HELLO count:* The total number of *HELLO* packets received from a neighbour node and (iv) *Ditch ratio*. The *distrust value* is calculated as per the formula (5). We modify the *HELLO* packets to include an additional field: (i) *Energy consumed*: Energy consumed by the node, which is provided as information to the neighbour nodes (calculated as per the formula (3)).

### 4.1. Routing strategy

We further modify the routing strategy described in Ref. [5]. The modified routing strategy (by modifying *Step 4*, *Step 8* and *Step 9*) is described herewith:

*Step 1*: Before starting data transmission, the source node *ns* looks up in its local routing table for the destination node $n_d$.

*Step 2*: If entry exists, it starts sending data through the trusted next hop to $n_d$. Go to *Step 8*.

*Step 3*: If no such route exists, $n_s$ initiates a route discovery process by flooding route request (RREQ) packets to discover a route to $n_d$.

*Step 4*: When an intermediate node $n_k$ receives a route reply (RREP) from its neighbour node $n_j$, it accepts the reply only if $n_j$ is not a distrusted node ($n_k$ finds absence of attack patterns for $n_j$ with distrust value less than or equal to $\eta$) and *recommended as a trusted node*.

*Step 5*: If multiple route replies are received after the route discovery process, a route entry for the route with the highest destination sequence number and trusted next hop is created for *nd* and inserted into the routing table of *ns*.

*Step 6*: If no such route is discovered, go to *Step 3.*

*Step 7*: Node $n_s$ starts data transmission to *nd*.

*Step 8*: If an intermediate node $n_k$ finds a next hop $n_m$ distrusted (*by direct observation or by recommendation*) in its routing table for a destination $n_p$ during the trust update procedure, the entry is discarded. A local route discovery process is initiated by $n_k$ to discover an alternate route to $n_p$.

http://dx.doi.org/10.5772/66519

*Step 9*: Even though an intermediate node *nk* finds a distrusted neighbor $n_m$ attempting to regain its trust by recuperating the distrust value less than or equal to $\eta$, *it is still considered as a distrusted node* (i.e. it is *not* reconsidered as a trusted node).

### 4.2. Routing procedures

The procedures for sending RREQ, receiving RREQ and sending RREP remain unmodified as presented in **Figures 1–3**, respectively (as described in Ref. [5]).

**Algorithm 1: *SendRREQ( )*** **//**By the source node

```
Fill up RREQ packet with the required fields
Broadcast the RREQ packet to discover route to the destination
```

**Figure 1.** *SendRREQ* procedure [5].

**Algorithm 2: *ReceiveRREQ( )*** **//**By the destination node or an intermediate node

```
If (The received RREQ is duplicate) then
   Discard the RREQ
Else
   If (New or updated route is found) then
      Update the routing table entry for the source node
      Construct or update reverse route towards the source node
   End If
   If (The receiving node is either the destination or intermediate node with fresher route) then
      SendRREP( )
   Else
      Record the required field values from the received RREQ for SL2
      Update necessary fields in the RREQ before rebroadcasting
      Rebroadcast the RREQ packet
   End If
End If
```

**Figure 2.** *RecvRREQ* procedure [5].

**Algorithm 3: *SendRREP( )*** **//**By destination/intermediate node having fresher route

```
If (Sending node is the destination node) then
    Increment the destination sequence number
End If
Fill up RREP packet with the required fields
Unicast the RREP packet on the reverse route towards the source
```

**Figure 3.** *SendRREP* procedure [5].

The modifications carried out in the receiving RREP procedure are highlighted in **Figure 4**.

**Algorithm 4:** ***ReceiveRREP( )*** **//**By intermediate node on the reverse route or source node

```
Record the required field values from the received RREP (or from the overheard RREP)
Insert the corresponding recorded values into SL1 and SL2
If (The neighbor sending RREP is marked as distrusted or recommended as distrusted) then
   Discard the RREP
Else
   If (New or updated route is found) then
      Update the routing table entry for the destination node
   End If
   If (Receiving node is the source node) then
      Discard the RREP
      Send data through the forward route if the route is fresher and next hop is trusted
   Else
      Forward the RREP packet on the reverse route towards the source
   End If
End If
```

**Figure 4.** *RecvRREP* procedure.

The procedure for route maintenance remains unmodified as presented in **Figure 5** (as described in Ref. [5]).

**Algorithm 5:** ***MaintainRoute( )*** **//**By each node

```
If (A link is broken) then
   If (The route is active and the destination is within the maximum hop limit) then
      Initiate local route repairing
   Else
      Carry out required updates in routing table
      Notify upstream nodes about the broken link by sending RERR containing unreachable destinations
   End If
End If
If (RERR is received) then
   Carry out required updates in the routing table
   If (Receiving node is the source node) then
      Re-initiate route discovery process
   Else If (The route is active and the destination is within the maximum hop limit) then
      Initiate local route repairing
   Else
      Rebroadcast the RERR packet
   End If
End If
```

**Figure 5.** *Route maintenance* procedure [5].

### 4.3. Trust update and trust recommendation procedures

The modifications carried out in the trust update and trust recommendation procedures are highlighted in **Figures 6** and **7**, respectively.

http://dx.doi.org/10.5772/66519

**Algorithm 6:** ***UpdateTrust( )*** **//**By each node

**For** (Each neighbor table entry)
**Do**
  Verify the existence of attack patterns from *SL1* and *SL2* of the neighbor
  Calculate *DTV* value of the neighbor
    **If** (The neighbor follows attack patterns or has $DTV > \eta$) **then**
      Mark the node as a distrusted node
    **Else If** (The neighbor ***is not marked as a distrusted node*** and recommended as a trusted node) **then**
      The node remains as a trusted node
    **End If**
**Done**
**For** (Each routing table entry)
**Do**
  Find the entry of the next hop from the neighbor table
  **If** (The next hop is found to be distrusted ***or recommended as distrusted*** in the neighbor table) **then**
    Discard the route containing the malevolent next hop
    Initiate a local route discovery process to discover an alternate route to the destination
  **End If**
**Done**

**Figure 6.** *Update trust* procedure.

**Algorithm 7:** ***RecommendTrust( )*** **//**By each node

//Before broadcasting a HELLO packet//
Construct an empty Blacklist for recommendation purpose
**For** (Each neighbor table entry)
**Do**
  **If** (The neighbor is marked as a distrusted node) **then**
    Insert the neighbor identity into the Blacklist
  **End If**
**Done**
Incorporate the Blacklist ***and Energy Consumption information*** into the HELLO packet
Broadcast the HELLO packet to the neighbors
//After receiving a HELLO packet//
Receive HELLO packet from the neighbor
***Enter the Energy Consumption information of the neighbor into the neighbor table***
**If** (The neighbor sending the HELLO packet is trusted) **then**
  Obtain the Blacklist from the HELLO packet
  **For** (Each entry in the Blacklist)
  **Do**
    Find the corresponding entry in the neighbor table
    **If** (The neighbor entry exists) **then**
      Set *Recommendation value* as 'distrusted' for the neighbor
    **End If**
  **Done**
**End If**

**Figure 7.** *Recommend trust* procedure.

## 5. Adversary models

It is obvious that the success of any security mechanism largely depends on the operations performed by the adversaries. In our work, we evaluate the performance of ETRS-PD against three adversary models described in Ref. [5].

### 5.1. Intelligent adversary model

The operations performed by *intelligent adversary* (denoted as *Attack1*) are presented in **Figure 8** [5, 20]. The adversary follows a pattern in inserting the value of hop count (Hop_Count = 2) while sending RREP packet.

**Procedure 1:** ***Actions by the malicious node to learn the routing information in promiscuous mode***

```
If (RREQ or RREP packet is received or overheard) then
   Update the information in the routing table
   If (Highest_Recorded_Dest_Seqno < Dest_Seqno in the received/overheard packet) then
      Highest_Recorded_Dest_Seqno = Dest_Seqno in the received/overheard packet
   End If
End If
```

**Procedure 2:** ***Actions by the malicious node after receiving an RREQ***

```
Discard the received RREQ
If (RREQ is NOT for me) then
   If (valid fresher route is available in the routing table) then
      Fill up RREP with Dest_Seqno=Incremented value of
                        Highest_Recorded_Dest_Seqno and Hop_Count=2
      Unicast the forged RREP on the reverse path to the source
   End If
Else
   Fill up RREP with own Seqno and Hop_Count=1
   Unicast the genuine RREP on the reverse path to the source
End If
```

**Procedure 3:** ***Actions by the malicious node after receiving a data packet from the source node***

```
If (data packet is NOT for me) then
   Time1=Receipt time of the first data packet
   Time2=time1+50% of the Total life time
   If (Time2 > Total life time) then
      Time2=Total life time
   End If
   If (Current Time ≥ time1 && Current Time ≤ time2) then
      Drop the data packet received from the source
   Else
      Forward the data packet
   End If
Else
   Receive the data packet for me
End If
```

**Figure 8.** Operations performed by a node launching *Attack1* [5, 20].

### 5.2. Slow poison adversary model

The operations performed by *slow poison adversary* (denoted as *Attack2*) are presented in **Figure 9** [5]. The adversary follows a pattern in inserting the value of destination sequence number (RREQ_Dest_Seqno + 1) while sending RREP packet.

**Procedure 1:** ***Actions by the malicious node after receiving an RREQ***

```
If (RREQ is NOT for me) then
   If (route is available in the routing table) then
      Fill up RREP with Dest_Seqno=RREQ_Dest_Seqno+1 and Hop_Count=Random(1,2)
      Discard the received RREQ
      Unicast the forged RREP on the reverse path to the source
   Else
      Discard the received RREQ
   End If
Else
   Fill up RREP with own Seqno and Hop_Count=1
   Unicast the genuine RREP on the reverse path to the source
End If
```

**Procedure 2:** ***Actions by the malicious node after receiving a data packet from the source node***

```
If (data packet is NOT for me) then
   Calculate the current time slot count i=int(Current_Time/10)
   Packets_to_be_dropped= Fibonacci(i)                    // f_i= f_i-1+f_i-2
   If (Total_packets_dropped_in_current_time_slot < Packets_to_be_dropped) then
      Drop the data packet received from the source
      Increment the value of Total_packets_dropped_in_current_time_slot
   Else
      Forward the data packet
   End If
Else
   Receive the data packet for me
End If
```

**Figure 9.** Operations performed by a node launching *Attack2* [5].

## 5.3. Capricious adversary model

The operations performed by *capricious adversary* (denoted as *Attack3*) are presented in **Figure 10** [5]. This adversary does not generate any attack pattern while sending RREP packet.

**Procedure 1:** ***Actions by the malicious node after receiving an RREQ***

```
Discard the received RREQ
If (RREQ is NOT for me) then
   If (valid fresher route is available in the routing table) then
      Fill up RREP with Dest_Seqno=Routing_table_Dest_Seqno+Random(1,7)
                        and Hop_Count=Random(1,3)
      Unicast the forged RREP on the reverse path to the source
   End If
Else
   Fill up RREP with own Seqno and Hop_Count=1
   Unicast the genuine RREP on the reverse path to the source
End If
```

**Procedure 2:** ***Actions by the malicious node after receiving a data packet from the source node***

```
If (data packet is NOT for me) then
   If (Packet_ID mod Random(1,3) == 0) then
      Drop the data packet received from the source
   Else
      Forward the data packet
   End If
Else
   Receive the data packet for me
End If
```

**Figure 10.** Operations performed by a node launching *Attack3* [5].

## 6. Simulation results and analysis

NS-2 (ver. 2.34) simulator is used to evaluate the performance efficiency of ETRS-PD against the three adversary models, namely *Attack1*, *Attack2* and *Attack3*. To prove our claim that ETRS-PD provides enhanced routing process than our previous proposal, TRS-PD [5], the performance of ETRS-PD is compared with TRS-PD against all three adversary models. We employ IEEE 802.11 MAC to carry out simulations in an area of 1000 × 1000 m. The benign nodes were randomly distributed over the network, which employs either AODV, ETRS-PD or TRS-PD protocol. Randomly positioned malicious nodes selectively perform packet forwarding misbehaviours by employing either of the three adversary models, namely *Attack1*, *Attack2* and *Attack3*. It is considered that the wireless network interface consumes 1.65, 1.4, 1.15 and 0.045 W for the *Transmit*, *Receive* and *Idle* modes and the *Sleep* state, respectively [21]. We take 800 μs as the transition time from the Sleep state to Awake state and during this transition period, a mobile node will consume 2.3 W power. All the experimental data are obtained after performing 10 different simulations and taking their average values. The major simulation parameters are shown in **Table 1**.

| Parameter | Value |
|---|---|
| Coverage area | 1000 × 1000 m |
| MAC layer protocol | IEEE 802.11 |
| Communication range of each node | 250 m |
| Channel bandwidth | 2 Mbps |
| Traffic type | CBR-UDP |
| Packet size | 512 bytes |
| Mobility model | Random way point |
| Simulation duration | 240 s |
| Number of nodes | 50 |
| Maximum mobility (varying) | 4–20 m/s |
| Pause time | 5 s |
| Number of connections | 15 |
| Percentage of malicious nodes (varying) | 0–40% |
| Routing protocols | AODV, *Attack1*, *Attack2*, *Attack3*, TRS-PD, ETRS-PD |
| Initial energy | 1000 J |
| Transmit power | 1.65 W |
| Receive power | 1.4 W |
| Idle power | 1.15 W |
| Sleep power | 0.045 W |
| Transition power | 2.3 W |
| Transition time | 800 μs |

**Table 1.** Simulation parameters.

In order to evaluate the performance of ETRS-PD, the following performance metrics are used: *packet delivery ratio (PDR)*, *normalized routing overhead (NRO)* and *average energy consumption (AEC)*. The following network parameters are varied: (1) *maximum speeds of nodes* and (2) *percentage of adversaries*.

The performance of AODV and TRS-PD in terms of PDR and NRO is already evaluated in Ref. [5], while their performance in terms of AEC is evaluated in Ref. [21].

### 6.1. Test 1: varying node mobility

In this test, the performance of the protocols is evaluated against *Attack1*, *Attack2* and *Attack3* by varying mobility of nodes from 4 to 20 m/s and keeping other parameters fixed. The percentage of malicious nodes is kept fixed to 20% for all three types of adversaries.

As shown in **Figure 11**, the PDR of AODV under *Attack1* declines from nearly 46 to 39% as the mobility increases from 4 to 20 m/s. The increase in packet loss at higher mobility is due to the increased number of link breakages at higher node speeds. Meanwhile, PDR of AODV under *Attack2* and *Attack3* declines from nearly 68 to 60% and 74 to 64%, respectively, as shown in **Figures 12** and **13**, respectively. When TRS-PD is employed, the PDR declines from nearly 73 to 57%, 79 to 69% and 80 to 69% under *Attack1*, *Attack2* and *Attack3*, respectively. This considerable rise in PDR is due to the integration of the attack-pattern discovery mechanism with the trust model. Meanwhile, when ETRS-PD is employed, it provides improvement in PDR over TRS-PD by an average of 6.21 under *Attack1*, 2.82 under *Attack2* and 4.03 under *Attack3*. The reasons behind improved results are as follows: (i) Construction of a composite trust metric using social trust and QoS trust. (ii) Enhanced routing decisions due to the modifications carried out in receive RREP, trust update and trust recommendation procedures.

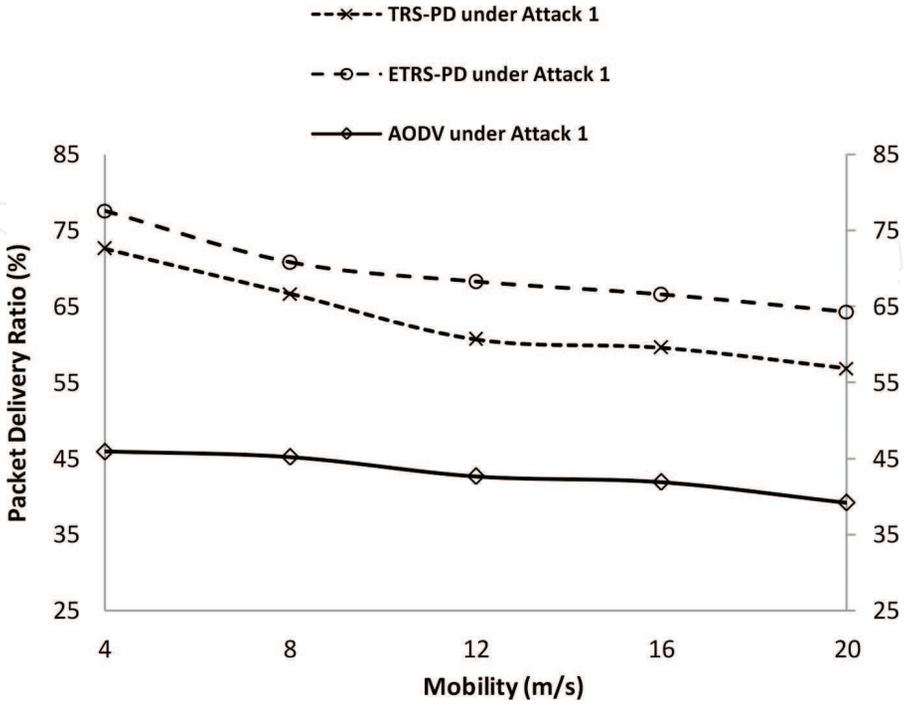


**Figure 11.** PDR under *Attack1*.

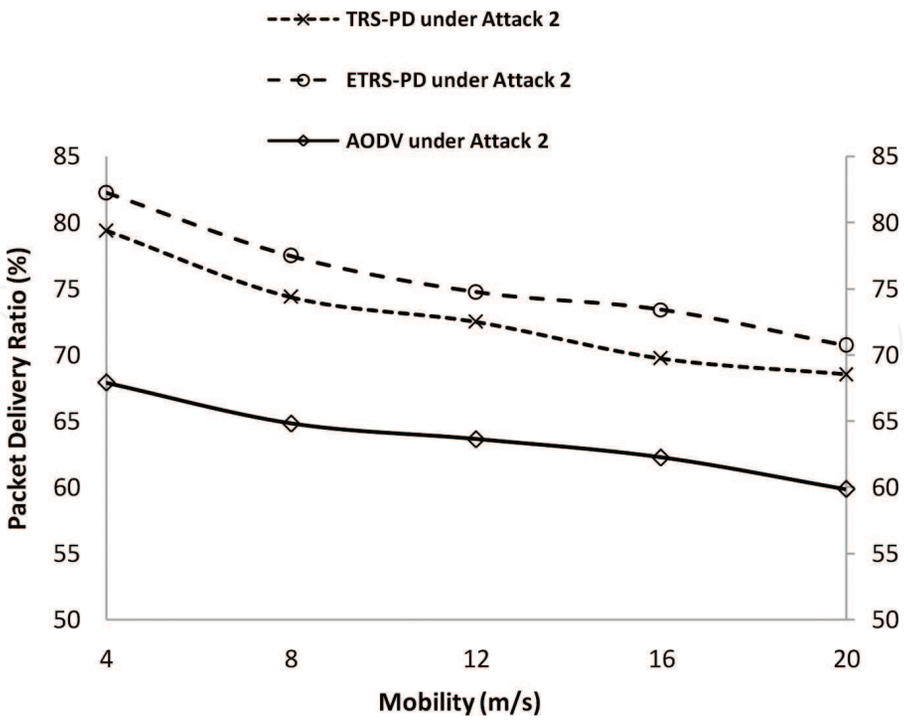


**Figure 12.** PDR under *Attack2*.

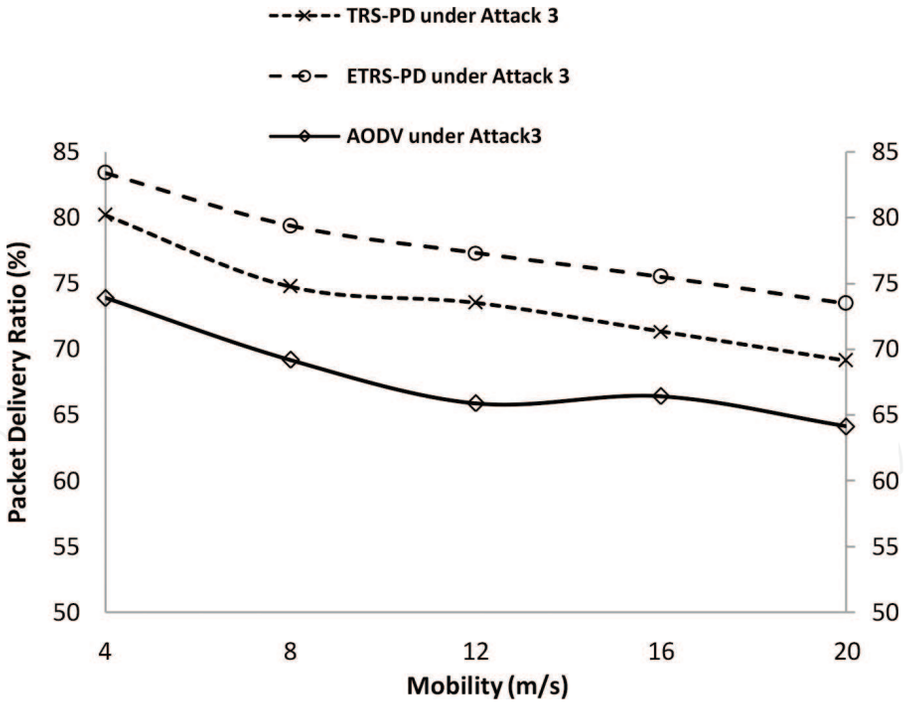


**Figure 13.** PDR under *Attack3*.

As shown in **Figures 14–16**, as the node speed increases, the NRO of AODV increases from nearly 5.7 to 10.4, 1.8 to 4.1 and 2.8 to 5.1 under *Attack1*, *Attack2* and *Attack3*, respectively.

http://dx.doi.org/10.5772/66519

Meanwhile, the TRS-PD provides improved performance over AODV by providing NRO from nearly 4.5 to 8.5 and 2.8 to 5.1 under *Attack1* and *Attack3*, respectively. On the other hand, due to the *Fibonacci dropping behaviour* of *Attack2* during the data transmission phase, the number of route hand-off mechanisms increases for TRS-PD as time goes on. As a result, resultant NRO is higher than that of AODV, which varies between nearly 3.2 and 5.5. Meanwhile, ETRS-PD provides improvement in NRO over TRS-PD by an average of 1.43 under *Attack1*, 0.30 under *Attack2* and 0.36 under *Attack3*. The reason behind this is, ETRS-PD leads to less number of route hand-off mechanisms than TRS-PD due to the inclusion of two more components in the overall trust composition as well as enhanced routing process.

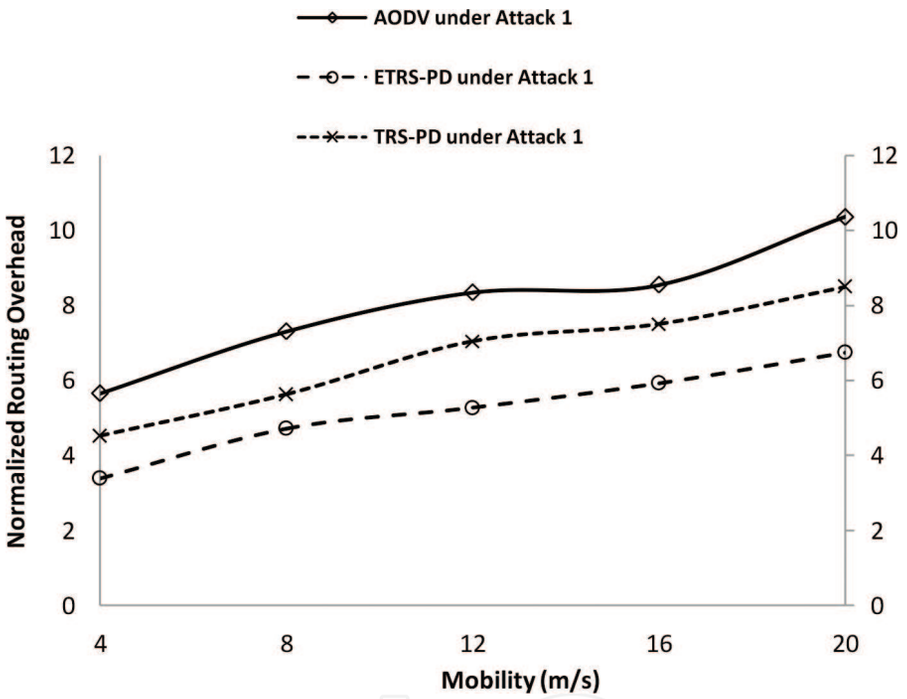


**Figure 14.** NRO under *Attack1*.

In order to ensure the improvement in energy consumption, we compare the performance of ETRS-PD with TRS-PD. As depicted by the graph in **Figure 17**, the AEC under *Attack1* varies between 313.56 and 314.13 J when employing TRS-PD. Meanwhile, ETRS-PD improves the AEC of TRS-PD by an average of 1.6 J. As depicted by the graph in **Figure 18**, the AEC under *Attack2* varies in the range of 312.82–314.4 J when employing TRS-PD. Meanwhile, ETRS-PD improves the AEC of TRS-PD by an average of 0.57 J. As depicted by the graph in **Figure 19**, the AEC under *Attack3* varies between 312.79 and 313.41 J when employing TRS-PD. Meanwhile, ETRS-PD improves the AEC of TRS-PD by an average of 0.57 J.

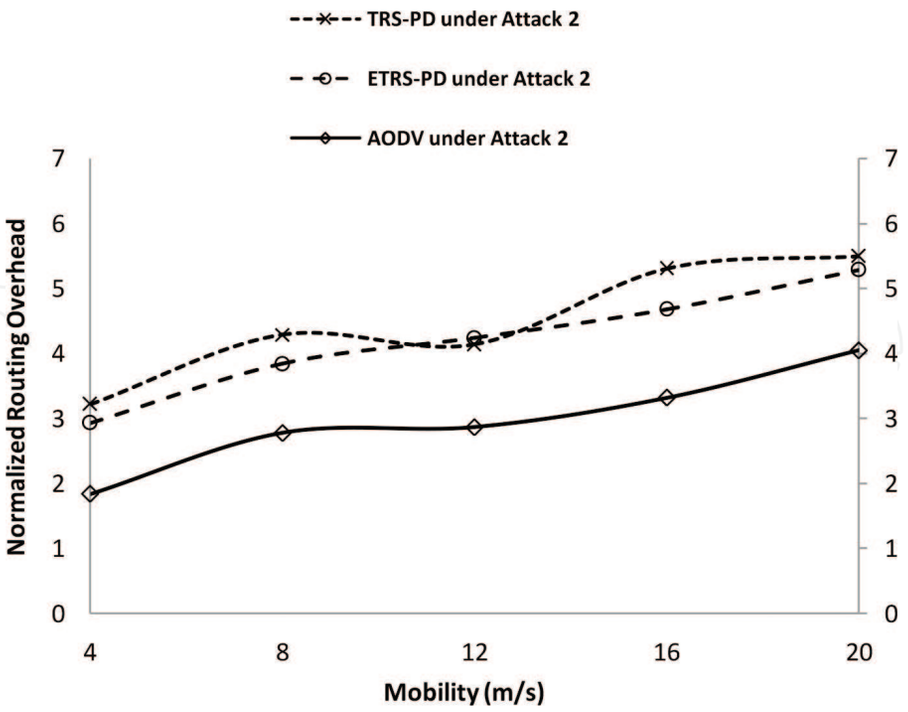


**Figure 15.** NRO under *Attack2*.

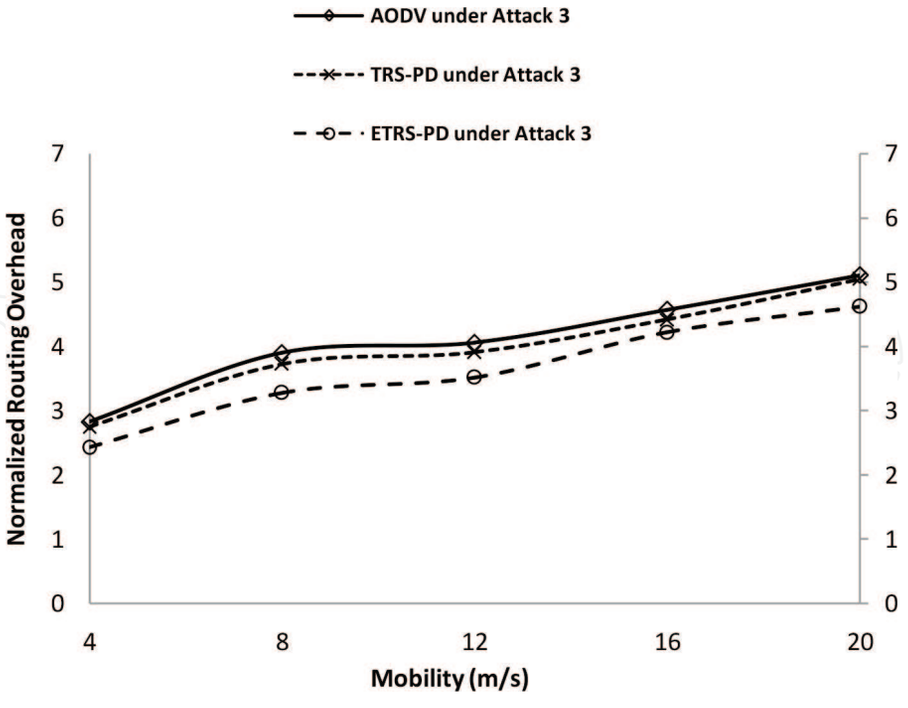


**Figure 16.** NRO under *Attack3*.

http://dx.doi.org/10.5772/66519

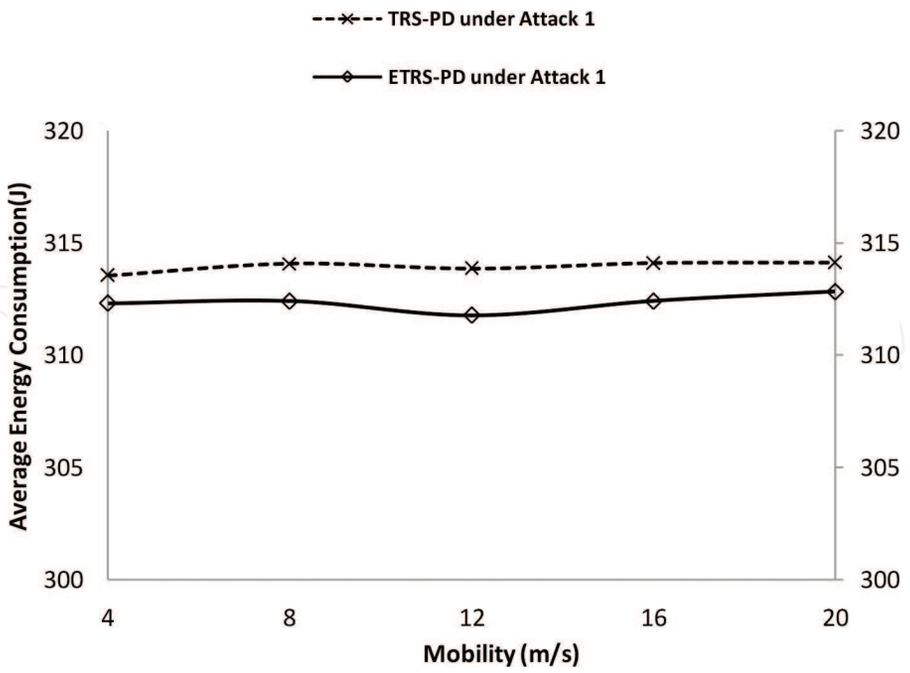


**Figure 17.** AEC under *Attack1*.

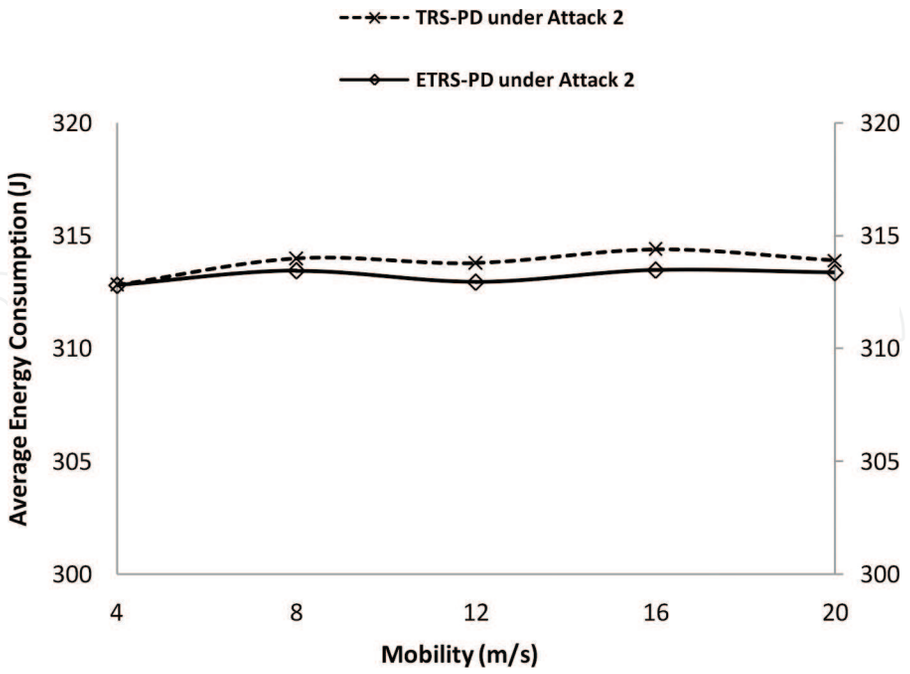


**Figure 18.** AEC under *Attack2*.

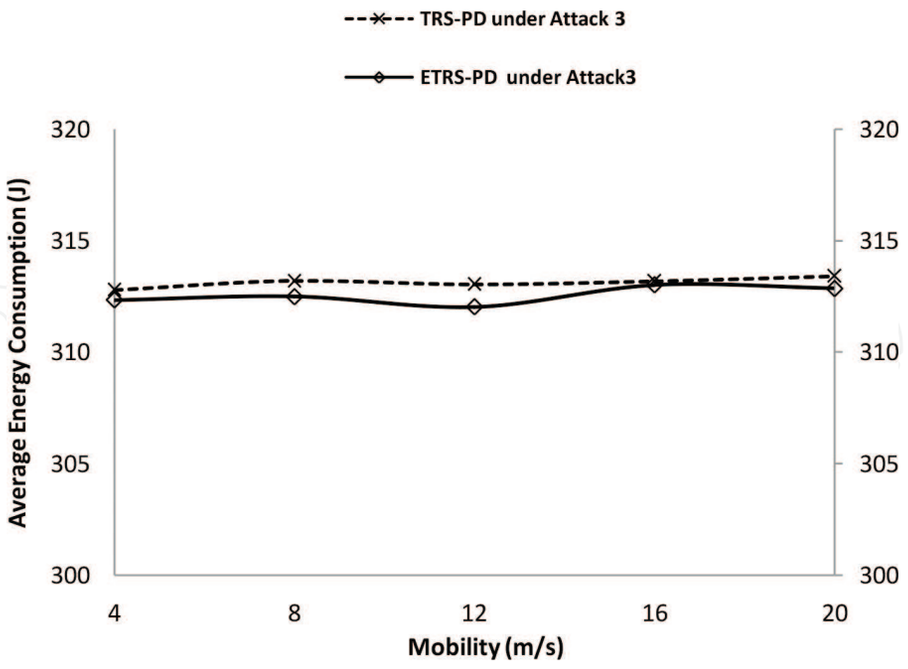


**Figure 19.** AEC under *Attack3*.

### 6.2. Test 2: varying percentage of malicious nodes

In this test, the performance of the protocols is evaluated against *Attack1*, *Attack2* and *Attack3* by varying percentage of malicious nodes from 0 to 40% and keeping other parameters fixed. The mobility parameter is kept fixed to 20 m/s for all three types of adversaries.

As shown in **Figures 20–22**, due to the increased intensity of packet dropping activities with the percentage increase in malicious nodes, the PDR of AODV declines from nearly 79 to 32%, 79 to 54% and 79 to 56% under *Attack1*, *Attack2* and *Attack3*, respectively. On the other hand, TRS-PD proves provides improvement in PDR of nearly 12 to 18%, 8 to 9% and 4.5 to 7% in the presence of malicious nodes launching *Attack1*, *Attack2* and *Attack3*, respectively. Meanwhile, in the presence of adversaries, ETRS-PD provides improvement in PDR over TRS-PD by an average of 7.67 under *Attack1*, 2.14 under *Attack2* and 4.29 under *Attack3*.

The NRO of AODV varies in the range of nearly 4.8–12.1, 3.9–4.8 and 4.7–5.4 under *Attack1*, *Attack2* and *Attack3*, respectively, as shown in **Figures 23–25**. On the other hand, TRS-PD improves NRO by maximum of 2.2 and 0.5 under *Attack1* and *Attack3* respectively over AODV. Meanwhile, TRS-PD increases NRO from nearly 0.7 to 2.0 under *Attack2* as compared to AODV. On the other hand, in the presence of adversaries, ETRS-PD provides improvement in NRO over TRS-PD by an average of 2.22 under *Attack1*, 0.25 under *Attack2* and 0.46 under *Attack3* due to the aforementioned reasons.

As shown in **Figures 26–28**, when employing TRS-PD, the AEC of the network without the presence of adversaries is 313.84 J while that is 312.35 J when employing ETRS-PD. As shown in **Figure 26**, the AEC for the MANET employing TRS-PD under *Attack1* varies between 314.08

http://dx.doi.org/10.5772/66519

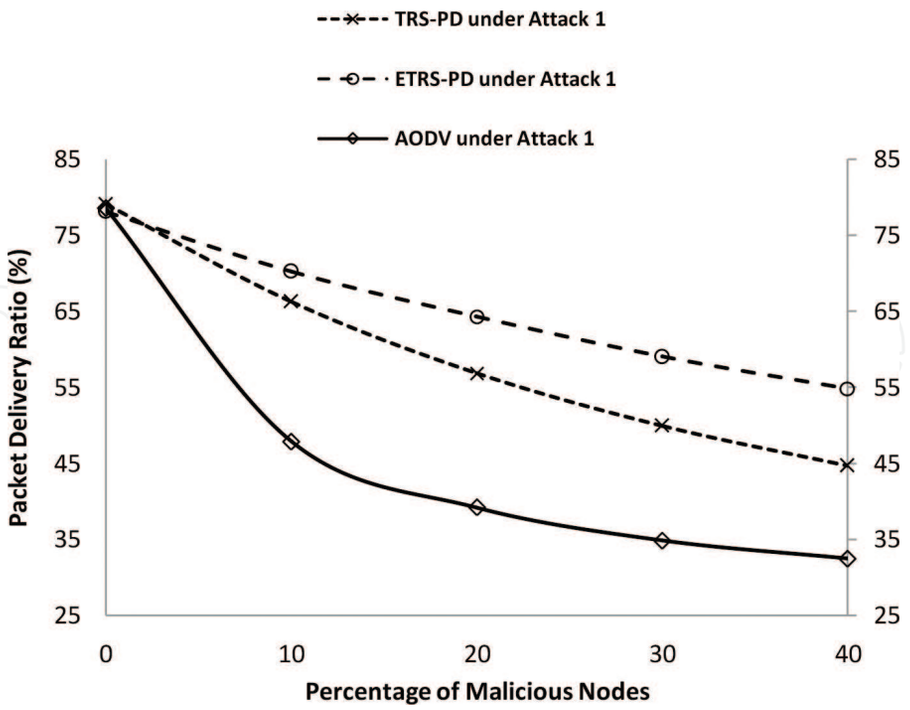


**Figure 20.** PDR under *Attack1*.

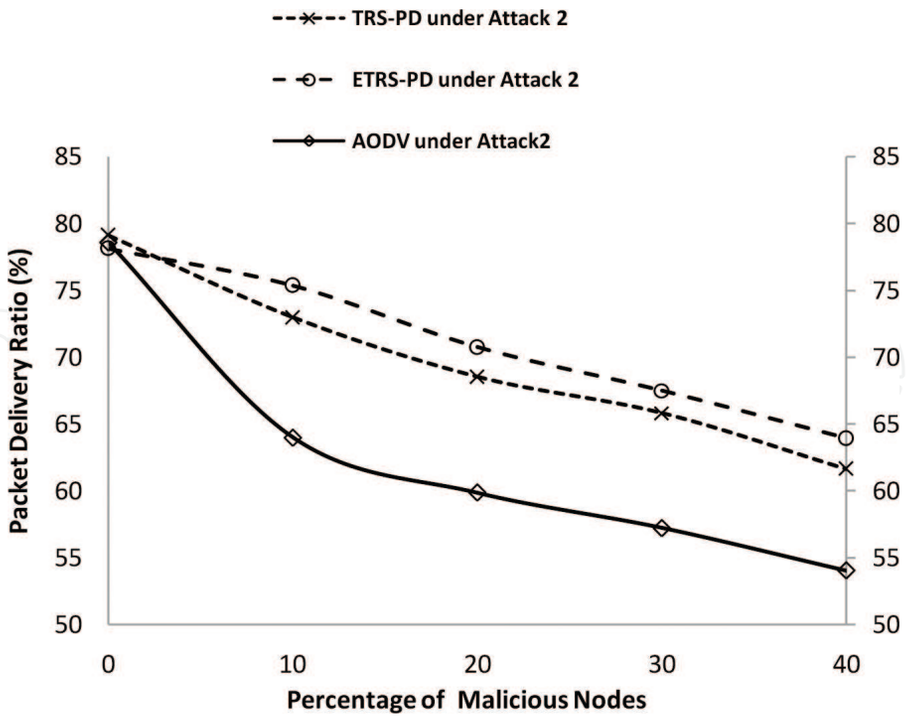


**Figure 21.** PDR under *Attack2*.

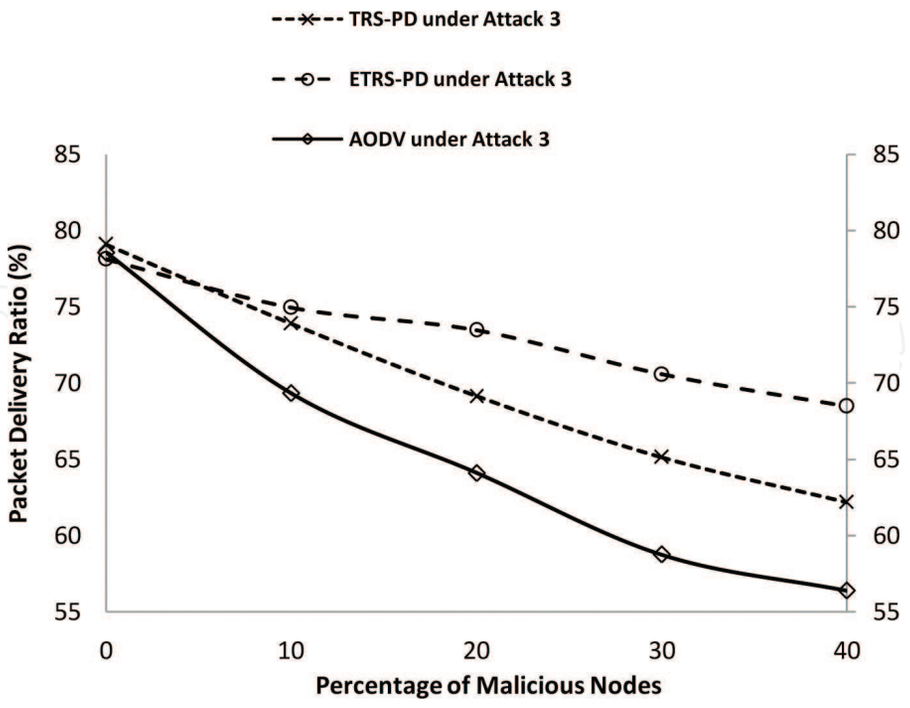


**Figure 22.** PDR under *Attack3*.

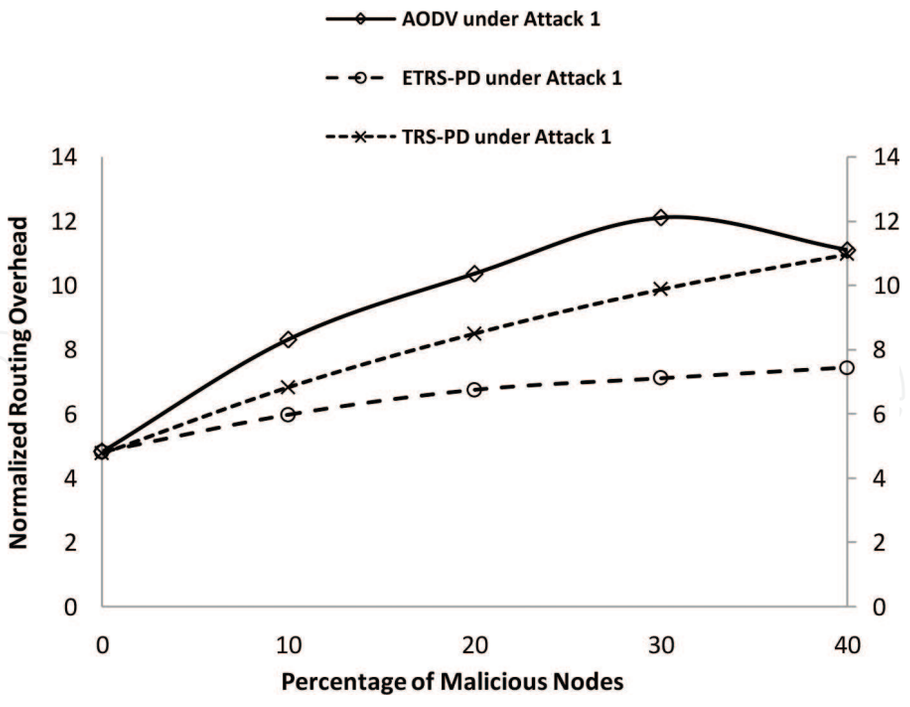


**Figure 23.** NRO under *Attack1*.

http://dx.doi.org/10.5772/66519

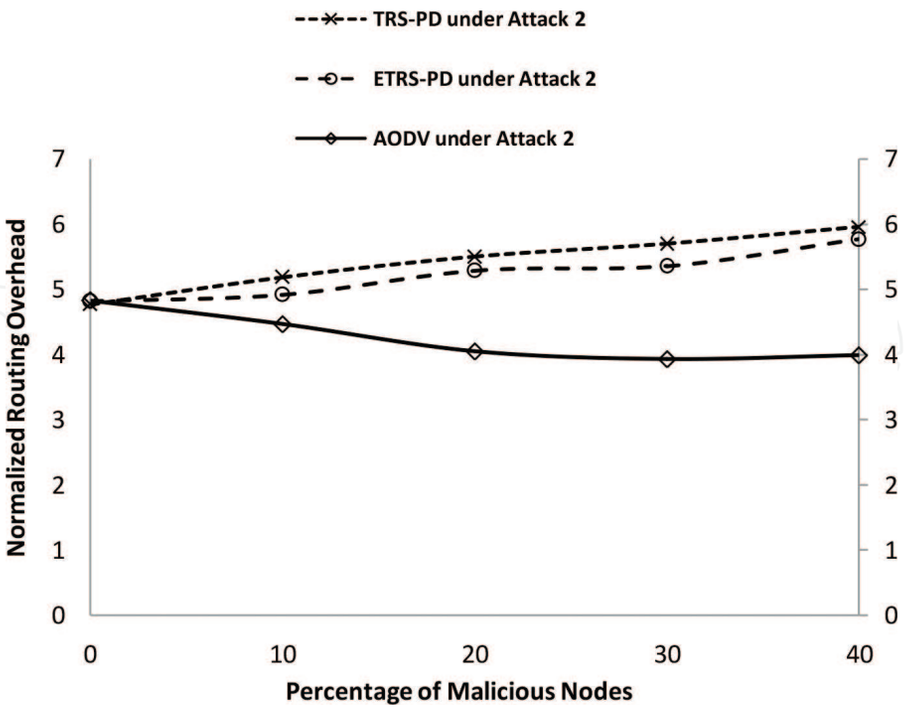


**Figure 24.** NRO under *Attack2*.

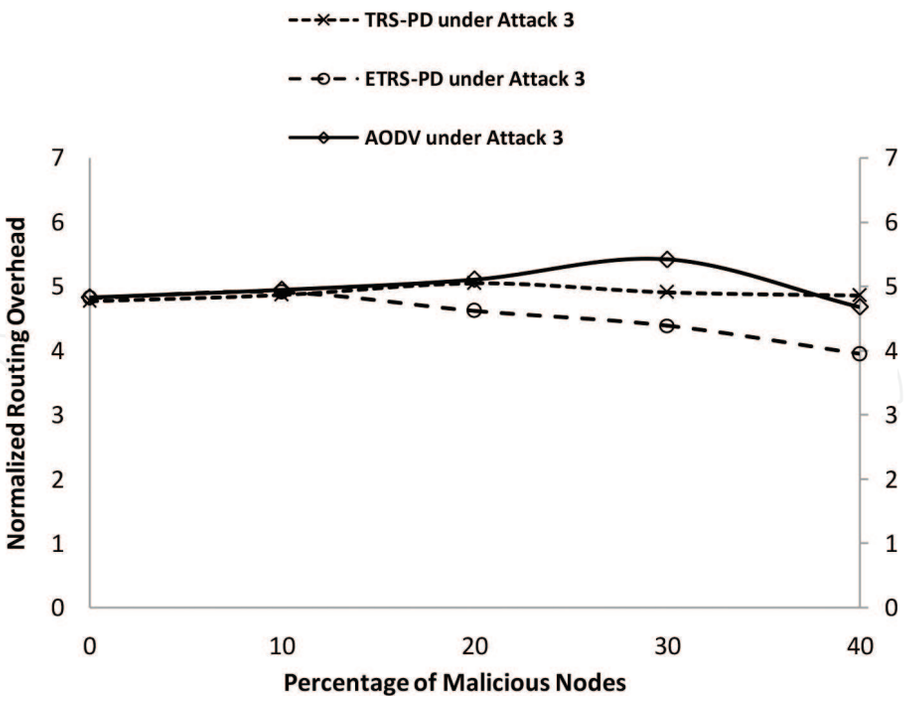


**Figure 25.** NRO under *Attack3*.

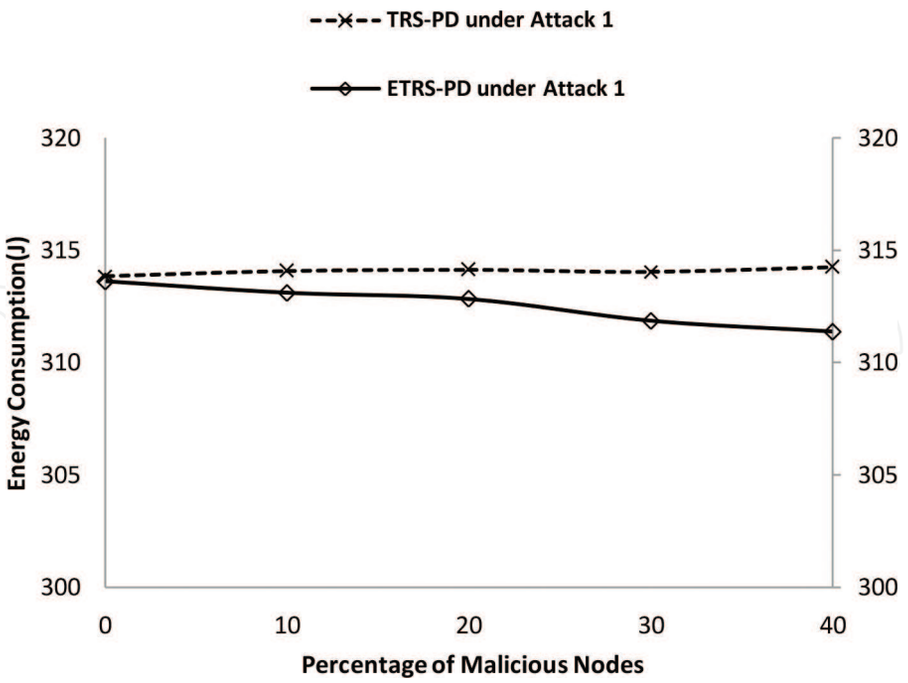


**Figure 26.** AEC under *Attack1*.

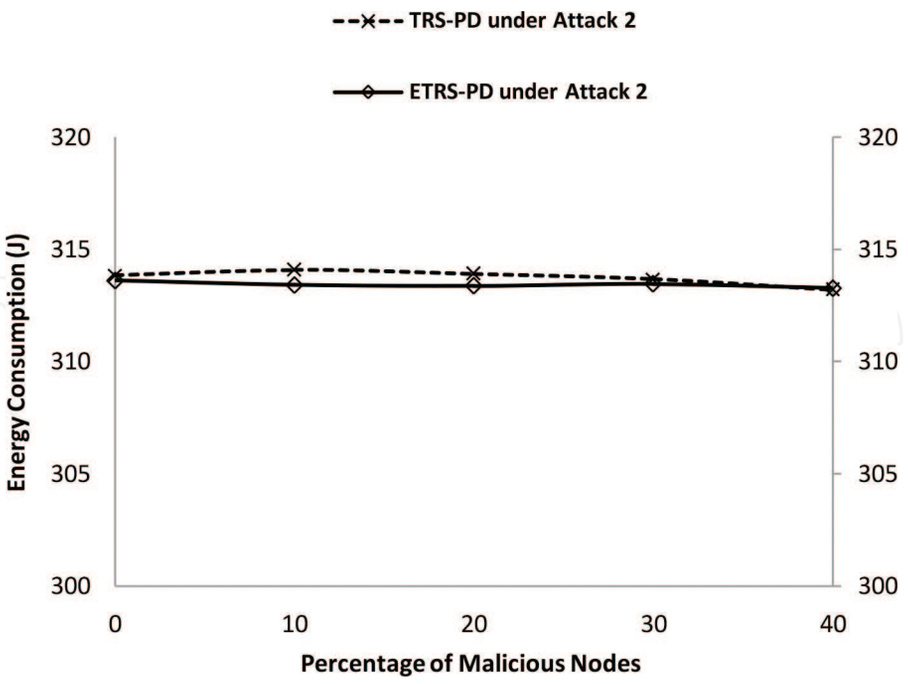


**Figure 27.** AEC under *Attack2*.

http://dx.doi.org/10.5772/66519

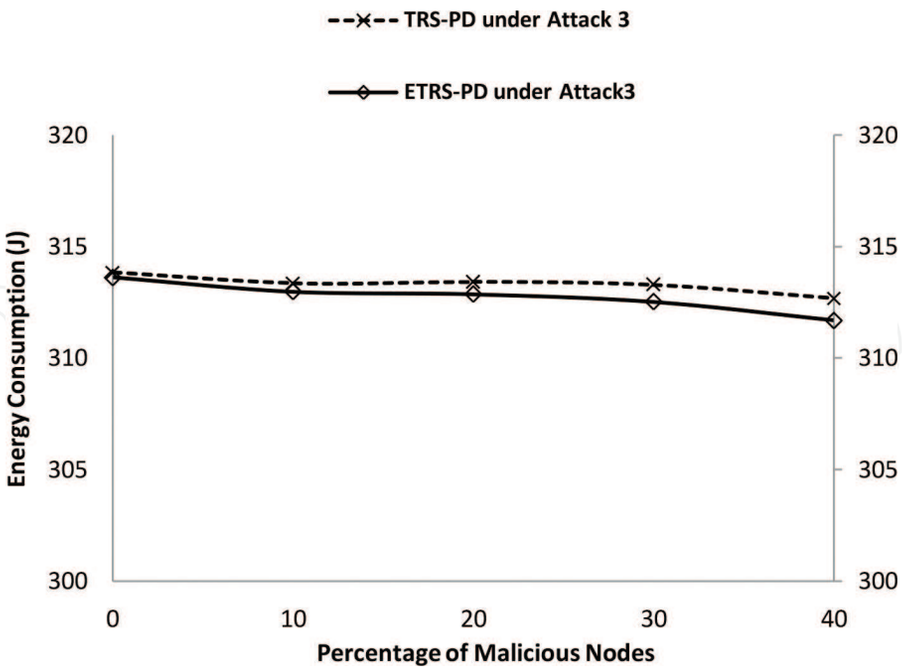


**Figure 28.** AEC under *Attack3*.

and 314.25 J. Meanwhile, ETRS-PD improves the AEC of TRS-PD by an average of 1.83 J in the presence of adversaries. As shown in **Figure 27**, the AEC of the MANET employing TRS-PD under *Attack2* decreases from 314.08 to 313.2 J. Meanwhile, ETRS-PD improves the AEC of TRS-PD by an average of 0.34 J in the presence of adversaries. As shown in **Figure 28**, the AEC of the MANET employing TRS-PD under *Attack3* varies between 312.68 and 313.41 J. Meanwhile, ETRS-PD improves the AEC of TRS-PD by an average of 0.67 J in the presence of adversaries.

## 7. Conclusions

As a part of the literature survey, we observe that integration of QoS trust and social trust in the composition of a trust metric would improve the performance of a trust-based scheme. Considering these notes, we modify our previous trust-based scheme, TRS-PD, such that it combines both the types of trust components. In addition, we suggest modifications in the route discovery, trust update and trust recommendation procedures of TRS-PD. The proposed trust-based approach, ETRS-PD, improves the routing decisions due to the suggested modifications. The performance comparison of ETRS-PD with TRS-PD under three distinct adversary models shows that ETRS-PD achieves remarkable improvement in packet delivery ratio due to the enhanced routing process and inclusion of two new trust components. Moreover, ETRS-PD reduces the generation of number of control packets due to the reduced number of route handoff mechanisms. As a result, ETRS-PD provides improved normalized routing overhead as well as energy consumption as compared to TRS-PD under different network scenarios.

## Author details

Rutvij H. Jhaveri[1]*, Narendra M. Patel[2] and Devesh C. Jinwala[3]

*Address all correspondence to: rhj_svmit@yahoo.com

1 SVM Institute of Technology, Bharuch, India

2 Birla Vishvakarma Mahavidyalaya, V.V. Nagar, India

3 Sardar Vallabhbhai National Institute of Technology, Surat, India

## References

[1] Junhai, L., Danxia, Y., Liu, X. and Mingyu, F. A survey of multicast routing protocols for mobile ad-hoc networks. IEEE Communications Surveys & Tutorials. 2009;**11**(1):78–91.

[2] Xia, H., Jia, Z., Ju, L., Li, X. and Sha, E.H.M. Impact of trust model on on-demand multi-path routing in mobile ad hoc networks. Computer Communications. 2013;**36**(9):1078–1093.

[3] Yu, H., Shen, Z., Miao, C., Leung, C. and Niyato, D. A survey of trust and reputation management systems in wireless communications. Proceedings of the IEEE. 2010;**98**(10):1755–1772.

[4] Marchang, N. and Datta, R. Light-weight trust-based routing protocol for mobile ad hoc networks. IET Information Security. 2012;**6**(2):77–83.

[5] Jhaveri, R.H. and Patel, N.M. Attack-pattern discovery based enhanced trust model for secure routing in mobile ad-hoc networks. International Journal of Communication Systems. 2016; DOI: 10.1002/dac.3148.

[6] Xia, H., Jia, Z., Li, X., Ju, L. and Sha, E.H.M. Trust prediction and trust-based source routing in mobile ad hoc networks. Ad Hoc Networks. 2013;**11**(7):2096–2114.

[7] Gharehkoolchian, M., Hemmatyar, A.A. and Izadi, M. Improving security issues in MANET AODV routing protocol. In: International Conference on Ad Hoc Networks; Springer International Publishing; 2015. pp. 237–250.

[8] Airehrour, D., Gutierrez, J. and Ray, S.K. Gradetrust: a secure trust based routing protocol for MANETs. In: International Telecommunication Networks and Applications Conference (ITNAC); IEEE; 2015. pp. 65–70.

[9] Patel, V.H., Zaveri, M.A. and Rath, H.K. Trust based routing in mobile ad-hoc networks. Lecture Notes on Software Engineering. 2015;**3**(4):318–324.

[10] Chiejina, E., Xiao, H. and Christianson, B. A dynamic reputation management system for mobile ad hoc networks. Computers. 2015;**4**(2):87–112.

[11] Mylsamy, R. and Sankaranarayanan, S. A preference-based protocol for trust and head selection for cluster-based MANET. Wireless Personal Communications. 2016;**86**(3): 1611–1627.

[12] Indirani, G. and Selvakumar, K. A swarm-based efficient distributed intrusion detection system for mobile ad hoc networks (MANET). International Journal of Parallel, Emergent and Distributed Systems. 2014;**29**(1):90–103.

[13] Xia, H., Yu, J., Tian, C.L., Pan, Z.K. and Sha, E. Light-weight trust-enhanced on-demand multi-path routing in mobile ad hoc networks. Journal of Network and Computer Applications. 2016;**62**:112–127.

[14] Azer, M.A. and Saad, N.G.E.D. A new reputation system for misbehavior detection and control in ad hoc networks. In: International Computer Science and Engineering Conference (ICSEC); IEEE; 2015. pp. 1–6.

[15] Rajkumar, B. and Narsimha, G. Trust-based light weight authentication routing protocol for MANET. International Journal of Mobile Network Design and Innovation. 2015;**6**(1):31–39.

[16] Cho, J.H., Swami, A. and Chen, R. A survey on trust management for mobile ad hoc networks. IEEE Communications Surveys & Tutorials. 2011;**13**(4):562–583.

[17] Cho, J.H., Swami, A. and Chen, R. Modeling and analysis of trust management for cognitive mission-driven group communication systems in mobile ad hoc networks. In: International Conference on Computational Science and Engineering; IEEE; 2009. pp. 641–650.

[18] Kohlas, R., Jonczy, J. and Haenni, R. A trust evaluation method based on logic and probability theory. In: IFIP International Conference on Trust Management; Springer US; 2008. pp. 17–32.

[19] Reidt, S., Wolthusen, S.D. and Balfe, S. Robust and efficient communication overlays for trust authority computations. In: Sarnoff Symposium (SARNOFF); IEEE; 2009. pp. 1–5.

[20] Jhaveri, R.H. and Patel, N.M. A sequence number based bait detection scheme to thwart grayhole attack in mobile ad hoc networks. Wireless Networks. 2015;**21**(8):2781–2798.

[21] Jhaveri, R.H. and Patel, N.M. Evaluating energy efficiency of secure routing schemes for mobile ad-hoc networks. International Journal of Next-Generation Computing. 2016;**7**(2):130–143.

IntechOpen