**TADL: A Trust-Aware Dynamic Location-Based Protocol Suite For Discovering Multiple Paths in MANETs**

Helen Bakhsh  
University of Manchester  
Manchester, UK  
bakhshhh@cs.man.ac.uk

Dr. Ning Zhang  
University of Manchester  
Manchester, UK  
nzhang@cs.man.ac.uk

Dr. Andy Carpenter  
University of Manchester  
Manchester, UK  
andy@cs.man.ac.uk

**ABSTRACT**

This paper describes the design and evaluation of a multiple path discovery protocol suite aimed at finding a set of most trust-worthy paths for routing higher priority traffic towards a destination node with minimum overhead costs. The protocol suite, called a the Trust-Aware Dynamic Location-based (TADL) protocol suite, has taken three measures to optimize the path discovery outcome: (a) it uses a directional approach to broadcast route request packets to a destination, (b) it adjust the size of the searching area dynamically in response to the underlying network conditions, and (c) it uses trust values to govern the selection of neighbours in path formation and to reduce the overhead incurred in trust estimation, we have adopted a direct trust model. Measures (a) and (b) are taken to minimise the level of broadcast traffic injected into the network in order to accomplish the path discovery task, and measure (c) is taken to ensure that the paths discovered are of best quality in terms of reliable packet deliveries. TADL is evaluated, using simulation, against the LAR protocol that has only implemented (a) (i.e. LAR only uses a location-based directional broadcast approach) under various network conditions. Our simulation study shows that TADL outperforms LAR in terms of reducing the routing overheads in the network. This overhead reduction can have a positive effect on QoS making the TADL more effective and efficient in providing QoS.

**Keywords**

Mobile ad hoc networks, MANETs, Quality of Service, QoS, Trust aware, Directional location, Multiple path

1. INTRODUCTION

A MANET [19] is a self-configuring wireless network that is formed dynamically by a collection of wireless mobile nodes. The nodes are typically mobile, resulting in dynamic changes in network topologies and fluctuations in network available bandwidth. A MANET does not have any infrastructural support, so to provide the network functionality, it relies on the collaborations among the network nodes themselves. This not only places additional processing and communication loads onto the communication nodes, but also opens up doors to more active attacks by intermediate nodes [8]. The additional loads and the active attacks have detrimental effects on Quality of Services (QoS). In addition, the open wireless media also make the network vulnerable to passive attacks. To protect against these attacks, security mechanisms are usually used, and these mechanisms impose additional computational and communication overheads, which, in turn, would deplete QoS. It is also worth noting that, mobile nodes are typically battery powered, and they are more restrictive in terms of storage space and processing capabilities than their wired counterpart [18]. These MANET features indicate that providing QoS in MANETs is a challenging issue. One of the challenging issues is how to find good or reliable paths to deliver higher priority traffic towards a destination node with minimum overhead costs. In addition, here is a trade-off between QoS and security and how to balance this trade-off is also a challenging issue. Therefore, it may be more cost effective to consider QoS and security in an integrated manner when designing a QoS solution for adversarial MANETs [9]. The security issues considered in this paper are packet dropping attacks or any misbehaviours in forwarding packets by intermediate nodes.

As our first step towards designing a cost-effective QoS solution in the presence of adversaries, this paper presents a novel route discovery protocol for discovering multiple paths with minimum bandwidth overheads in the presence of node mobility and packet dropping attacks. This protocol, called the Trust-Aware Dynamic Location-based (TADL) route discovery protocol, has the following features. Firstly, to minimize any unnecessary broadcast traffic poured into the underlying network during a route discovery process, it uses a destination-location-based route discovery approach to discover paths. With this approach, route request packets are only broadcast within a defined zone, called searching area, with the source and destination nodes located at the edges of the area. Secondly, to cater for dynamic changes in network conditions (e.g. node density levels, node mobility levels, topology changing levels, etc) and in packet dropping ratios, it dynamically changes the size of the searching area. It is hoped that, in this way, we can discover an optimal number of paths with minimum bandwidth overhead costs. Thirdly, to thwart packet dropping attacks by intermediate nodes, the protocol assesses and uses nodes trust values to govern the selection of nodes in path constructions. To reduce the overhead incurred in this process, the protocol uses...
a direct trust model. It is hypothesized that, by using the most trusted paths, the probability of successfully delivering high priority traffic could be maximized.

We use simulation to evaluate the effectiveness of the TADL approach. Our simulation results show that TADL is effective in reducing the number of control packets injected into the underlying network and it also decreases the average end-to-end packet delivery delays. In addition, TADL outperforms the most relevant related work, the LAR protocol, in terms of decreasing the average end-to-end packet delivery delay and increasing the packet delivery ratio.

The rest of this paper is organised as follows. Section 2 reviews and critically analyses related works. Section 3 gives the preliminaries for the TADL design. Section 4 describes the design of the TADL, TADL components and how it is used to find multiple most reliable paths. Section 5 reports the simulation study of the protocol. Finally, section 6 concludes the paper and outlines our future work.

2. RELATED WORKS

Existing efforts on supporting QoS can largely be divided into two groups, one using a reservation-based approach, and the other using a non-reservation-based approach. With the reservation-based approach [5, 20, 14], the bandwidth required by a higher priority traffic flow is reserved along the path between the source and the destination prior to the delivery of the traffic flow. However, in a highly mobile network where the network topology changes dynamically, a reservation may become void before the data delivery is completed, and this should happen, a new reservation along a new path would have to be made. These repeated path reservations will lead to an increased number of reservation packets being injected into the underlying network, and these packets will consume bandwidth and deplete QoS. In addition, reservation packets themselves are also targets of attacks. Protecting them will impose a higher computational burden on the communication nodes, and possibly increase the bandwidth overhead as well, both of which will have a negative effect on QoS.

With the non-reservation-based approach, efforts have been focusing on designing routing protocols to reduce packed delivery delays and/or to increase the reliability of successful packet delivery in the presence of limited bandwidth resources, node mobility and/or security attacks. Examples of these routing protocols are the multi-path routing protocols [10] and the location-based routing protocols [3]. The multi-path routing protocols are designed to discover multiple paths between a given source and destination pair in a single route discovery process. One of the discovered multiple paths is then used as the primary path, and the others are used as secondary or backup paths. In an event when the primary path fails, data can still be delivered over another path [2]. In this way, packets can be delivered with less delay. The multi-path routing approach may also be used to provide load balancing or to harvest bandwidths over multiple paths to speed up packet delivery. In this case, instead of using the multiple paths in a sequential order, each packet is fragmented into multiple blocks, and the blocks are delivered using more than one path simultaneously [4]. Also, this approach may be more resistant against packet dropping attacks. In a situation where the attacker ratio is high, data can be duplicated and sent through multiple paths [17]. Study results [11, 10] have shown that, under certain conditions such as attacker ratio is high, using multi-path routing in ad hoc networks can lead to a better performance than using single path routing. However, there are also disadvantages in using multi-path routing. Maintaining multiple paths to a destination requires each node to maintain a larger routing table and to make path selections from multiple paths discovered imposes additional node complexity on source nodes.

With location-based routing protocols, network nodes position information is used to restrict the flooding of route discovery packets. The route discovery packets are only forwarded towards the direction of a destination node, rather than broadcast across the entire network. In this way, the bandwidth used by the route discovery packets (i.e. the control overhead) can be reduced, which may help to achieve a better QoS. Example of these protocols are Location-Aided Routing (LAR) [3], Predictive Location-Based QoS Routing (PLBQR) [15] and QoS Greedy Perimeter Stateless Routing (QoS GPSR) [1].

The above-mentioned QoS solutions are mostly designed under the assumption that intermediate nodes are trustworthy and cooperative in routing packets. Security related contributions in QoS solution designs are mostly focused on protecting the authenticity of data and/or control packets. However, the attacks coming from misbehaving insiders can have more severe impacts on QoS. To tackle routing unreliability caused by intermediate nodes, there have been works on designing solutions that make QoS routing decisions based on intermediate nodes trust values. There are two types of trust models proposed in literature, a global trust model and a local trust model. With the global trust model [21], each node in a network needs to know the trust value of every other node in the network. This is typically achieved by having nodes to exchange trust relevant information over the network. Exchanging information at the whole network scale can introduce a high level of traffic overheads and even cause congestions in the network. Additionally, the trust values conveyed may not be truthful; they may be faked or altered during transit. On the other hand, with the local trust model [13], each node only needs to learn the trust values of their immediate neighbours. They learn these values by direct experiences or interactions with, and observations of, their neighbouring nodes. In comparison with the global trust model, this local trust model based approach generates less traffic overheads into the network and may impose less computational overheads onto network nodes, as measures applied to protect the trust values carried in the packets can be avoided.

As discussed above, most related works in the area of MANET QoS and security are carried out separately. The idea of addressing the two issues in an integrated manner was first mentioned in the literature by McNerney and Zhang in their work [9]. In [9], they proposed to use packet duplication over multi-paths to reduce the impact of packet dropping attacks on QoS. However, the issue of how to discover multiple paths with minimum bandwidth overheads introduced into the underlying network was not addressed in their work. The TADL routing protocol, to be described in the remaining part of this paper, is designed to address this issue.

3. TADL: HIGH-LEVEL IDEAS

The TADL protocol is designed with the following criteria
in mind: (a) it should be able to discover multiple paths between a pair of source and destination nodes with minimum bandwidth costs; (b) while discovering paths, it should be able to select the most reliable nodes to form the paths so as to optimise packet delivery ratios in later stage, i.e. in data forwarding stage, and do so with minimum communication overhead and computational complexity on the intermediate nodes. For criterion (a), three ideas are used. The first is to use a directional approach to multi-path discovery, i.e. the route discovery packets are forwarded within a searching area bounded by the source and destination nodes, rather than flooding the whole network regardless of the locations of the two nodes. In this way, we can reduce the level of broadcast traffic injected into the network, thus reducing the bandwidth overheads. But we needed to find a way to define the destination node location and the searching area. The second idea is to adjust the size of the searching area dynamically in response to the underlying network conditions. Different network conditions may impact on the number of paths that can be discovered or used. For example, if the network density level is high, there will be more paths available. In this case, we can afford to reduce the size of the searching area. Conversely, if the density level is low, then there will be fewer paths discoverable, and in this case, the size of the searching area should be larger. The network attacker ratio should also have an influence on the number of paths that should be discovered. The higher the attacker ratio, the more paths we should discover and use, as to increase delivery reliability, we may need to duplicate packet transmission over more paths. To optimize the trade-off between the number of paths that can be discovered and the bandwidth overheads consumed by route discovery packets, TADL dynamically adjust the searching area in which route discovery packets are broadcast. The third idea is to send the route request, during the route discovery process, to some of its neighbouring nodes [6] rather than broadcast the request packets inside the searching area. Those neighbouring nodes are the most trusted neighbours in the searching area of the destination node.

For criterion (b), TADL uses trust values to govern the selection of neighbours in path formation and to reduce the overhead incurred in trust estimation, it uses a direct trust model [13, 7]. This is because, as discussed earlier, the direct trust model does not require trust value exchanges on the whole network scale, thus cost less in terms of communication overhead and computational complexity on the intermediate nodes than the global trust model. These measures are taken to ensure that better quality paths are discovered with minimum overhead costs.

4. TADL: LOW-LEVEL DESIGN

This section describes the design of TADL in detail. It gives detailed coverage as how the ideas and measures described above are implemented in the design of this protocol to realize our vision of trust-aware dynamic location-based multi-path discovery. Before the protocol description, we first spell out the assumptions and notations used in the design.

4.1 Assumptions

The following assumptions are used in the TADL protocol design.

A.1 The channel properties, including the trust values, are associated to each channel, i.e., may not identical in the two opposite directions between two nodes. In node A, the trust value of node B is not the same as node A trust value in node B.

A.2 With regard to security threats, data dropping attacks and any misbehaviours in forwarding packets are considered. Data dropping attacks are only mounted on data packets. In other words, they only drop data packets and forward control packets. The justification of this is that attackers, with the purpose of dropping data packets, want to be included in paths, so they must participate in route discovery process. Where as the misbehaviours in forwarding packets can affect the data and control packets, i.e., overloaded nodes are misbehave nodes that drops data and control packets.

A.3 Source and destination nodes trust each other. Attackers are only from intermediate nodes.

4.2 Notations

Table 1 shows the notations used in the TADL protocol design.

4.3 TADL Components

TADL consists of a number of components, which collectively perform the functions defined for TADL. These components are the Routing Table (RT), the Neighbouring Node Information (NNI) table, the Neighbouring Node Trust Value Estimation (TVE) method, the Neighbour Selection Algorithm (NSA), the Adjusting Searching Area Algorithm (AS2A) and a paths Selection Algorithm (PSA). This section describes these components.

4.3.1 Routing Table (RT)

Each node in the network maintains a RT that contains information to determine how to forward the data and control packets. It is used to store network nodes physical information and paths to the that node. Network nodes physical information is maintained during control and data packets transmission. Each node adds its physical location to any packet they send or re-send. Upon receiving a packet, the receiver node will add/update the corresponding physical location attribute value from the value carried in the that packet. The physical information attributes include the network node physical location measured as $X_i, Y_i$, where $X_i$ is node $N_i$ physical location in the x coordinate and $Y_i$ is node $N_i$ physical location in the y coordinate, the node mobility speed measured in meters per second and the time when the node information was collected/updated.

The path(s) is constructed when a source node needs to communicate with another node. The path(s) discovered before starting a communication session and maintained during the session. After the session, the routing entry will be expired and deleted within a specific expiring time, only up-to-date paths will be stored. The path contains information about all the intermediate nodes within the path to reach that node. Each node entry in the list contains the node physical information and the node trust value measured with an integer from the range of $[TRV_{min}, TRV_{max}]$, where $TRV_{min}$ is the minimum trust value, $TRV_{max}$ is the maximum trust value. The trust value of the nodes within the path is used to determine the path trust value.
### 4.3.2 Neighbouring Node Information (NNI) Table

Each node in the network maintains an NNI table storing attributes value for each of its directly connected neighbours. The attributes are the physical location and the trust value of the neighbouring nodes. Each node periodically broadcasts a HELLO packet to its neighbours containing the node current physical location and mobility speed. Upon receiving the HELLO packet, the receiving node adds or updates its NNI table with the received information. If the node does not receive a HELLO packet from a particular neighbour \(N_i\) within a specified time period, the node would consider neighbour \(N_i\) as unreachable and delete its record from the table.

#### Table 1: Notations Used in Protocol Description

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>Source node</td>
</tr>
<tr>
<td>D</td>
<td>Destination node</td>
</tr>
<tr>
<td>V</td>
<td>Node moving speed</td>
</tr>
<tr>
<td>R</td>
<td>Radius of a node expected zone</td>
</tr>
<tr>
<td>((X_i, Y_i))</td>
<td>Physical location of node (N_i)</td>
</tr>
<tr>
<td>(P_{ji})</td>
<td>The (ji)-path connecting between a source and destination nodes</td>
</tr>
<tr>
<td>(L_{ji})</td>
<td>Link connecting node (N_i) to its neighbour (N_j)</td>
</tr>
<tr>
<td>(N_{ji})</td>
<td>Neighbour node (N_i)</td>
</tr>
<tr>
<td>TRV</td>
<td>Trust value</td>
</tr>
<tr>
<td>(TRV_{ps})</td>
<td>Trust value for path (P_s)</td>
</tr>
<tr>
<td>(TRV_{rij})</td>
<td>Trust value for the link linking node (j) to (i)</td>
</tr>
<tr>
<td>(TRV_{min})</td>
<td>Minimum trust value</td>
</tr>
<tr>
<td>(TRV_{max})</td>
<td>Maximum trust value</td>
</tr>
<tr>
<td>CTR</td>
<td>Control packets</td>
</tr>
<tr>
<td>(CTR_{suc})</td>
<td>Number of delivered control packets</td>
</tr>
<tr>
<td>(CTR_{fail})</td>
<td>Number of failed control packets</td>
</tr>
<tr>
<td>(CTR_{all})</td>
<td>Number of sent control packets</td>
</tr>
<tr>
<td>DAT</td>
<td>Data packets</td>
</tr>
<tr>
<td>(DAT_{suc})</td>
<td>Number of delivered data packets</td>
</tr>
<tr>
<td>(DAT_{fail})</td>
<td>Number of failed data packets</td>
</tr>
<tr>
<td>(DAT_{all})</td>
<td>Number of sent data packets</td>
</tr>
<tr>
<td>OBS</td>
<td>Packets transmission direct observation value</td>
</tr>
<tr>
<td>(OBS_{L_{ji}})</td>
<td>Direct observation value of the link (L_{ji}) reliability</td>
</tr>
<tr>
<td>(OBS_{ctr})</td>
<td>Control packets direct observation value</td>
</tr>
<tr>
<td>(OBS_{dat})</td>
<td>Data packets direct observation value</td>
</tr>
<tr>
<td>SAR</td>
<td>Searching area size</td>
</tr>
<tr>
<td>(SAR_{cur})</td>
<td>Current searching area size</td>
</tr>
<tr>
<td>(SAR_{prev})</td>
<td>Previous searching area size</td>
</tr>
<tr>
<td>T</td>
<td>Time</td>
</tr>
<tr>
<td>(T_0)</td>
<td>Initial instance of time</td>
</tr>
<tr>
<td>(T_{prev})</td>
<td>Previous instance of time</td>
</tr>
<tr>
<td>(T_{cur})</td>
<td>Current instance of time</td>
</tr>
<tr>
<td>(T_{recv})</td>
<td>Data packet receiving time</td>
</tr>
<tr>
<td>(T_{trans})</td>
<td>Data packet transmission time</td>
</tr>
<tr>
<td>THR</td>
<td>Threshold value</td>
</tr>
<tr>
<td>(THR_{thr})</td>
<td>Trust value threshold</td>
</tr>
<tr>
<td>(THR_{ph})</td>
<td>Number of paths threshold</td>
</tr>
<tr>
<td>(THR_{att})</td>
<td>Attacker ratio threshold</td>
</tr>
<tr>
<td>(THR_{mob})</td>
<td>Mobility level threshold</td>
</tr>
<tr>
<td>TRR</td>
<td>Time threshold</td>
</tr>
<tr>
<td>NPS</td>
<td>Number of paths</td>
</tr>
<tr>
<td>(NPS_{prev})</td>
<td>Previously discovered number of paths</td>
</tr>
<tr>
<td>(NPS_{req})</td>
<td>Currently required number of paths</td>
</tr>
<tr>
<td>(W_{dat})</td>
<td>Weight assigned to data packets</td>
</tr>
<tr>
<td>(W_{ctr})</td>
<td>Weight assigned to control packets</td>
</tr>
<tr>
<td>NRQ</td>
<td>Number of selected neighbors</td>
</tr>
<tr>
<td>(DL_{ave})</td>
<td>Average end-to-end packet delivery delay</td>
</tr>
<tr>
<td>PDR</td>
<td>Packet delivery ratio</td>
</tr>
<tr>
<td>THR</td>
<td>Trust value threshold</td>
</tr>
<tr>
<td>(THR_{mob})</td>
<td>Mobility level threshold</td>
</tr>
<tr>
<td>TRR</td>
<td>Time threshold</td>
</tr>
<tr>
<td>NPS</td>
<td>Number of paths</td>
</tr>
<tr>
<td>(NPS_{prev})</td>
<td>Previously discovered number of paths</td>
</tr>
<tr>
<td>(NPS_{req})</td>
<td>Currently required number of paths</td>
</tr>
<tr>
<td>(W_{dat})</td>
<td>Weight assigned to data packets</td>
</tr>
<tr>
<td>(W_{ctr})</td>
<td>Weight assigned to control packets</td>
</tr>
<tr>
<td>NRQ</td>
<td>Number of selected neighbors</td>
</tr>
<tr>
<td>(DL_{ave})</td>
<td>Average end-to-end packet delivery delay</td>
</tr>
<tr>
<td>PDR</td>
<td>Packet delivery ratio</td>
</tr>
</tbody>
</table>

### 4.3.3 Neighbouring Node Trust Value Estimation (TVE) Method

As mentioned earlier, one of the tasks each node ought to perform in TADL is to estimate a trust value for each of its neighbours. This is done by using the TVE method. The method uses a local trust model, i.e. each node estimates a neighbouring node trust value based on the quality of the link connecting to the neighbouring node. In other words, the trust value assigned to node \(N_i\) actually reflects the reliability of the link linking this node and node \(N_j\).

Assuming that the trust value a node \(N_j\) assigns to node \(N_i\) at current instance is denoted as \(TRV_{T_{cur}}^{L_{ji}}\). Initially when node \(N_i\) newly joins the neighbourhood of node \(N_j\), node \(N_j\) assigns a neutral value to \(TRV_{T_{0}}^{L_{ji}}\), which is calculated using Equation 1.

\[
TRV_{T_{0}}^{L_{ji}} = \frac{TRV_{min} + TRV_{max}}{2}
\]  

Where \(TRV_{min}\) is the minimum trust value, \(TRV_{max}\) is the maximum trust value and \(T_0\) indicates the initial instance of time when node \(N_i\) join the neighbouring node of node \(N_j\). This neutral value (calculated using Equation 1) indicates that node \(N_i\) cannot yet determine whether the link connecting to this new neighbour is trustworthy or not. Then, from this point on, node \(N_j\) updates the value of \(TRV_{T_{cur}}^{L_{ji}}\) regularly based on the current instance direct observation value of the reliability of the link, \(L_{ji}\), \(OBS_{T_{cur}}^{L_{ji}}\). \(OBS_{T_{cur}}^{L_{ji}}\) is calculated based on the number of acknowledgment packets successfully received from neighbour node, \(N_i\), at the interval from \(T_{prev}\) to \(T_{cur}\). That is, it is calculated as:

\[
OBS_{T_{cur}}^{L_{ji}} = (W_{ctr} \cdot OBS_{T_{cur}}^{L_{ji}}) + (W_{dat} \cdot OBS_{T_{cur}}^{L_{dat}})
\]  

Where \(OBS_{T_{cur}}^{L_{ctr}}\) and \(OBS_{T_{cur}}^{L_{dat}}\) are, respectively, the current instance direct observations of the link reliability, in terms of control and data packets, and \((W_{ctr} and W_{dat})\) are the weightings assigned to \((OBS_{T_{cur}}^{L_{ctr}} and OBS_{T_{cur}}^{L_{dat}})\), respectively.

\(OBS_{T_{cur}}^{L_{ctr}}\) is calculated as:

\[
OBS_{T_{cur}}^{L_{ctr}} = \frac{CTR_{suc} - CTR_{fail}}{CTR_{all}}
\]  

Where \(CTR_{suc}\) is the number of control packets that are sent by node \(N_i\) to node \(N_j\) at the interval from \(T_{prev}\) to \(T_{cur}\) and successfully acknowledged by node \(N_j\). The control packets include route request, route reply and route error packets. \(CTR_{fail}\) is the number of control packets that are sent by node \(N_j\) to node \(N_i\) at the same interval but failed to be acknowledged by node \(N_j\). \(CTR_{all}\) is the total transmitted control packet by node \(N_j\) to node \(N_i\) at the interval from \(T_{prev}\) to \(T_{cur}\); it is the sum of \(CTR_{suc}\) and \(CTR_{fail}\).

Similarly, \(OBS_{T_{cur}}^{L_{dat}}\) is calculated as:

\[
OBS_{T_{cur}}^{L_{dat}} = \frac{DAT_{suc} - DAT_{fail}}{DAT_{all}}
\]  

This OBS value indicates the reliability of the link (and node \(N_i\)) in successfully delivering/acknowledging the control packets from node \(N_j\) to node \(N_i\) at the current instance. If the link is less reliable and/or the receiving node, \(N_i\), less likely to respond, the OBS value will be lower, indicating...
From the above equations, it can be seen that we calculate the OBS values for control packets and data packets separately, and we assign different weightings to them respectively. This is because we have observed from literature [12] that attackers may exhibit different behaviors to control and data packets. The more important packet type should be assigned with a weight value that is higher than the weight values assigned to less important. For example, in data dropping attacks, attackers forward control packets and only drop data packets. In this case, data packets are considered more important than control packets and have a higher value than \( W_{dat} \), and the sum of these values, i.e. \( (W_{ctrl} + W_{dat}) \), should be 1.0.

Depending on the value of \( OBS_{ij} \), node \( N_j \) updates the value of \( TRV_{L_{ij}} \), based on Equation 5.

\[
TRV_{L_{ij}} = \begin{cases} 
TRV_{L_{ij}} + \Delta TRV \cdot OBS_{L_{ij}} \geq THR_{obs} \\
TRV_{L_{ij}} - \Delta TRV \cdot OBS_{L_{ij}} < THR_{obs} 
\end{cases}
\]

Where \( \Delta TRV \) is an increment value and \( THR_{obs} \) is a threshold value. These values are configurable based on the need of applications and the network environment.

4.3.4 Neighbours Selection Algorithm (NSA)

The NSA is used to select NRQ neighbours with the highest trust values. Once selected, the current node will send the Route REQuest (RREQ) packet to these neighbours. As shown in Algorithm 1, a node ranks its neighbouring nodes based on their link trust values, \( TRV_{L_{ij}} \), and then selects NRQ neighbours with the higher link trust values among the nodes that are in the destination node searching area. The searching area is a rectangular area that starts from the current location of \( S \) and includes the predicted location of \( D \), as shown in Figure 1.

The predicted location of \( D \) is a circular region around \( D \). This region is the predicted location of \( D \) at time \( T_{cur} \) based on the known physical location and speed of \( D \) at time \( T_{prev} \). The center of the circle is \( D \) known physical location, \((X_d, Y_d)\), at time \( T_{prev} \). The radius \((R)\) of this circle is calculated as follows:

\[
R = V_{T_{prev}} \cdot (T_{cur} - T_{prev}) \cdot SAR_{T_{cur}}
\]

Where \( V \) is \( D \) speed at time \( T_{prev} \), \( T_{prev} \) is the time when this node received the lastest information update about \( D \), \( T_{cur} \) is the current time, and \( SAR \) is a value indicating the searching area starting size (Area-S, Area-M or Area-L). The determination of the \( SAR \) values and the AS2A algorithm, are explained in the next section.

Algorithm 1 NSA Algorithm

1: procedure NSA(SAR)
2:  Rank neighbours based on \( TRV_{L_{ij}} \)
3:  \( n \leftarrow 0 \)
4:  \( i \leftarrow 0 \)
5:  \( R = V_{T_{prev}} \cdot (T_{cur} - T_{prev}) \cdot SAR_{T_{cur}} \)
6:  Determine D searching area using \( X_d \), \( Y_d \), \( R \)
7:  \textbf{do}
8:    \textbf{if} \( N_i \) is in D searching area \textbf{then}
9:      SelectedNeighbours[\( n \)] \leftarrow N_i
10:    \textbf{end if}
11:  \( i \leftarrow i + 1 \)
12:  \textbf{while} \( n < NRQ \)
13:  \textbf{return} SelectedNeighbours
14: \textbf{end procedure}

4.3.5 Adjusting Searching Area Algorithm (AS2A)

To reduce bandwidth overheads incurred in a route discovery process, TADL allows a source node to dynamically adjust the size of the searching area in response to the underlying network condition and its QoS requirement. This is facilitated by specifying three distinctive sizes of the search area (Figure 2), small (Area-S), medium (Area-M) and large (Area-L) and an algorithm (i.e. AS2A) that implements a set of rules to govern the selection of these sizes.
pected zone circle around the destination node, so the SAR value for Area-M is 2. Doubling the size of the expected zone increases the area in which RREQ packets are broadcast (i.e., searching area) and this may lead to more paths being discovered. Area-L is the largest searching area that covers the entire network. In other words, when Area-L is used, RREQ packets will be broadcast to the entire network, de-faulting to the use of the flooding route discovery protocol.

From Area-S, to Area-M and then to Area-L, the size of the searching area progressively increases, the number of RREQ packets poured into the network also progressively increases, and more paths may be discovered. Of course, this also implies that more overheads will be generated. To optimize the trade-off between the number of paths that may be discovered and the level of broadcast traffic that has to be poured into the network to discovered the paths, we have designed a novel AS2A algorithm that governs which one of the three searching area sizes (Area-S, Area-M, or Area-L) the route discovery protocol should start with it in a given route discovery instance (one instance is one RREQ send, and hereafter referred to as one search). Intuitively, this choice may be made based on the number of discovered paths in the previous RREQ instance and the required number of paths as determined by the QoS requirement by the source node. For example, if the number of discovered paths in the previous RREQ instance is equal to the required amount of paths, the current search may start from the same searching area size used in the previous search. If the number of paths discovered in the last search is higher than the required number of paths, then the SAR value used in the current search may be the value used in the previous search decremented by 1. If the number of paths discovered in the previous search is less than the current required number of paths, the SAR value used in the current search should be incremented by 1.

However, the above decision rules may be adequate if the underlying network is static, which is typically not true in a mobile and hostable MANET environment. There are three further factors which should be considered when choosing a searching area size to start a route discovery instance. These factors are the network nodes attacker ratio, the neighbouring nodes average mobility level, and the time when the previous instance of RREQ was sent.

**Factor 1 - neighbouring nodes average attackers ratio:** This is the ratio of the number of attackers over the total number of nodes, in the network. When the attacker ratio in the network is high, the possibility of finding trustworthy paths in a small searching area is low. In this case, AS2A will start the search from the largest searching area, Area-L, which covers the whole network, and the probability of finding a sufficient number of good paths is higher.

**Factor 2 - neighbouring nodes average mobility level:** This is the average mobility level of the neighbouring nodes. If the average mobility level is higher, the network topology will change more frequently and the paths discovered in the previous occasion would be more likely to become invalid. In other words, the network topology would likely be different in each route discovery instance. In such a case, to minimize bandwidth overheads, AS2A starts with the smallest searching area, Area-S.

**Factor 3 - time when the previous route discovery was carried out:** The topology is not only affected by the node mobility, but it also changes with time. The longer the time elapses since the last route discovery operation, the more likely the topology will change. Therefore, we have specified a threshold value for the time it elapses since the last route discovery operation. If $T_{prv}$ is above this threshold value, the paths discovered in the previous route discovery operation is considered as obsolete, and even if the number of previously discovered paths is sufficient versus the required number of paths, AS2A still starts from the smallest searching area, Area-S.

The above-discussed rules have been implemented into an algorithm, called adjusting search area (AS2A) algorithm (shown in Algorithm 2).

**Algorithm 2 AS2A Algorithm**

```
1: procedure AS2A
2: if AttackRatio >= THR_{att} then \triangleright Use Area-L
3: SAR ← 3;
4: else
5: if MobilityLevel >= THR_{mob} then \triangleright Use Area-S
6: SAR ← 1;
7: else
8: if T_{cur} - T_{prv} >= THR_{T} then \triangleright Use Area-S
9: SAR ← SAR_{prv};
10: else
11: if NPS_{prv} = NPS_{req} then \triangleright Area size does not change
12: SAR ← SAR_{prv};
13: else
14: if NPS_{prv} > NPS_{req} then \triangleright Decrement area size
15: SAR ← SAR_{prv} - 1;
16: else
17: SAR ← SAR_{prv} + 1; \triangleright Increment area size
18: end if
19: end if
20: end if
21: end if
22: return SAR
23: end procedure
```

### 4.3.6 Paths Selection Algorithm (PSA)

The PSA algorithm is used after the route discovery process to select the $NPS_{req}$ most trusted paths that will participate in the traffic forwarding among the discovered paths. As shown in Algorithm 3, $S$ will calculate the trust value, $TRV_{px}$, of each of the discovered path, The $TRV_{px}$ is calculated using the minimum value of the trust values over all the links along that path. This calculation is based on the weakest link principle. Based on the trust values, $S$ ranks the paths and selects $NPS_{req}$ paths from the top of the list.

**Algorithm 3 PSA Algorithm**

```
1: procedure PSA(DiscoveredPaths)
2: Calculate TRV_{px} of each path
3: Rank the paths base on TRV_{px}
4: Select the best NPS_{req} paths
5: return NPS_{req}SelectedPaths
end procedure
```
4.4 Route Discovery Using TADL

This section describes the process by which TADL is used to discover and select multiple paths, i.e., a route discovery process using TADL. In this process, a source node executes a route discovery algorithm (RDA) that implements the methods and algorithms described in the section above and searches the network to find a required number of paths (denoted as $NPS_{req}$) linking the source node $S$ to a destination node $D$ in the direction of $D$. As shown in Algorithm 4, RDA operates as follows. Given a required number of paths, $NPS_{req}$, which is typically determined as part of the QoS requirements, the source node, $S$, first searches its routing table to see if there are already $NPS_{req}$ valid paths towards the destination node, $D$. If yes, $S$ uses these paths to transmit the high priority traffic. Otherwise, $S$ reads $D$ physical location from its routing table. If $S$ does not know $D$ location, $S$ defaults to use the basic broadcast method to discovery paths. In this case, the source node $S$ and all the intermediate nodes that receive the RREQ packet will broadcast the RREQ packet to all the neighbouring nodes, until the destination node is reached. However, if $S$ knows $D$ location, it selects NRQ most trusted neighbouring nodes in the direction of $D$ and sends a RREQ packet to these neighbours. The NRQ most trusted neighbouring nodes are selected using the NSA algorithm that are explained in Section 4.3.4.

Algorithm 4 RDA Algorithm For Source Node

```
1: procedure RDA For Source Node($NPS_{req}$)
2: Search routing table for $NPS_{req}$ paths to $D$
3: if $NPS_{req}$ paths to $D$ is found then
4:   SelectedPaths ← PSA(DiscoveredPaths)
5:   Transmit traffic over SelectedPaths
6: else
7:   Search the routing table for $(X_d,Y_d)$
8:   if $(X_d,Y_d)$ is not found then
9:      SelectedNeighbours ← AllNeighbours
10: else
11:    do
12:      SAR ← AS2A()
13:      SelectedNeighbours ← NSA(SAR)
14:      Send RREQ to the SelectedNeighbours
15:      Timer ← 0
16:    do
17:      Timer ← Timer + 1
18:      Wait for a RREP(s) from $D$
19:      while Timer < RDTimer
20:     if $NPS_{req}$ RREP(s) arrived then
21:        SelectedPaths ← PSA(DiscoveredPaths)
22:        Transmit traffic over SelectedPaths
23:    else
24:      if SAR = SAR$_L$ then
25:        $D$ is an unreachable destination
26:      end if
27:      end if
28:      while SAR != SAR$_L$
29:      end if
30:    end if
31: end procedure
```

Upon the transmission of a RREQ packet, node $S$ initiates a route discovery timer (RDTimer). By the expiration of this timeout interval, if $S$ does not receive a sufficient number of Route REPLY (RREP) packets, a new RREQ will be transmitted in a larger searching area than the one just used (as explained in the AS2A algorithm in Section 2). If, by the expiration of RDTimer, $S$ receives all, or a sufficient number, of RREP packets, $S$ will start the PSA that are explained in Section 4.3.6 to select NRQ paths from the discovered paths. $S$ then sends the data traffic via the selected paths to the destination node.

An intermediate node, upon receiving a RREQ packet, appends the value of the RREQ attributes associated to this intermediate node in the RREQ header. These attributes are this intermediate node ID and physical location, trust value of the next selected neighbour. It then forwards the RREQ packet on to its neighbouring nodes selected with the NSA algorithm. Intermediate nodes do not reply to the RREQ packet back to the source node. Rather, they forward the RREQ packet towards the direction of the destination node until the destination node is reached.

When the destination node receives the RREQ packet, it constructs a route reply packet, RREP, and records its current location, speed, current time and the entire path carried in the RREQ header into the RREP header. The destination node then send the RREP to the source node via the reverse of the path.

5. SIMULATION STUDY

In this section, we present the results of our simulation study of the TADL protocol suite and compare the results against those from LAR [3], the route discovery protocol most relevant to TADL.

5.1 Performance Metrics

The simulation study is carried out using three performance metrics, routing overhead, packet delivery ratio and average end-to-end packet delivery delay. The definitions of these metrics are given below:

Routing Overhead [4]: This is the total number of control packets transmitted by all the nodes in the network divided by the total number of data packets received at destination nodes.

\[
RoutingOverhead = \frac{CTR_{all}}{DAT_{rec}} (7)
\]

Packet Delivery Ratio (PDR): This is the total number of data packets received divided by the total number of data packets transmitted.

\[
PDR = \frac{DAT_{rec}}{DAT_{all}} (8)
\]

Average End-to-End Packet Delivery Delay (DLY$_{2e}$): This is the time difference between when a data packet is transmitted by the source node $T_{trn}$ and when the data packet arrives at the destination node $T_{rec}$.

\[
DLY_{2e} = T_{rec} - T_{trn} (9)
\]

5.2 Simulation Model

The study is carried out using the network simulator NS-2 (version 2.27.2) [16]. The network consists of 50 nodes located in a 1000m x 1000m area. Nodes move according to the Random Waypoint Mobility model. They move with...
a minimum speed of 1 m/s, and a maximum speed of 19 m/s. The duration of each simulation run is 900 seconds. Constant bit rate (CBR) traffic is transmitted at a rate of 4 packets per second. The packet size used is 512 byte. 30% of the source nodes transmit high priority traffic, whereas, the remaining 70% of source nodes transmit low priority traffic. These parameter settings are commonly used in the relevant studies published in literature [9].

5.3 Simulation Results

The performance of TADL has been investigated using a simulation study and the results are compared against LAR, the most relevant protocol to TADL. The study shows the effects of varying levels of traffic load, attacker ratio, and network node mobility on routing overheads, packet delivery ratios and average end-to-end packet delivery delays. The traffic load used is transmitted by 20%, 40%, 60%, 80%, and 100% of the nodes. Attackers are carried out by 0% - 30% of the nodes in the networks in 5% increments. Attackers only drop all/some data packets; they correctly forwards control packets. Different pause times are used to reflect the mobility level. These are 0, 300, 600, and 900 seconds. The LAR traffic (marked LAR) and the TADL traffic (marked TADL) are plotted in charts. Results are averaged over 30 simulation runs.

In the first set of results, Figures 3(a), 3(b) and 3(c), we examine the effects of traffic loads, attacker ratios, and network node mobility levels on routing overheads. From those figures, we can make a number of observations. Firstly, when the network load increases, routing overheads increase steadily with both TADL and LAR protocols. This observation is in line with our expectation - more traffic means more packet transmissions, which requires more control packets to discover more paths to deliver them. What is significant is that, for the whole range of network loads investigated, TADL outperforms LAR. On average, it reduces the routing overheads by 50.59%. As shown in Figure 3(a), when the traffic load is 100% (i.e. under the heavy traffic load condition), the routing overhead generated by TADL is 34.8% lower than LAR. This is a significant improvement in terms of routing overhead reduction, indicating that, when the pause time is 300s, attacker ratio is 0%, to discover a given number of paths, resizing the searching area dynamically can significantly reduce the number of control packets injected into the underlying network.

Secondly, as shown in Figure 3(b), for both protocols, an increase in the attacker ratio in the network has very little effect on the routing overheads generated effect. This observation tells us that the source node does not do much in terms of re-initiating path discoveries in the presence of packet dropping attacks. Intuitively, if there is an attacker along a path, it would be desirable for the source node to avoid that path, which means that the source node should discover/use an alternative path(s) should any attack occurs. This further indicates that there should be a mechanism to notify the source node of any attacking events. Figure 3(c) shows the routing overheads versus node mobility level changes. From the results, we can see that, except when mobility level is the highest (i.e. 0 second pause time is used), TADL significantly outperforms LAR, with an average routing overhead reduction of 78.3%. More interestingly, the results show that LAR gives the worst performance when the mobility level is the lowest (i.e. when a 900 second pause time is used, and with this pause time, the network nodes are largely stationary), and TADL gives the worst performance when the mobility level is the highest (i.e. when a 0 second pause time is used and with this pause time, the network nodes move continuously), with routing overheads overtaking those from LAR.

TADL differs from LAR in three aspects: (1) TADL dynamically adjusts the size of the search area in which route discovery packets are broadcast, whereas LAR uses two size zone Area-S and Area-L, LAR start the search with Area-S and if there is no RREP found it then increase to Area-L, and (2) TADL selects and uses paths which are more trustworthy, the trust value of a path is calculated based on the trust values of the links forming the path, whereas LAR does not have this path selection functionality, LAR uses the first discovered path. (3) In LAR, the intermediate node only passes the RREQ once, filters out any duplicated RREQ, i.e., after receiving each RREQ, the intermediate node has to check the received RREQ sequence number, as each RREQ has a unique sequence number, to distinguish it from other RREqs, the intermediate node will accept the RREQ only once and will discard any other received copies of the same RREQ. Whereas, in TADL, the intermediate node allows the RREQ to passes through with no extra checks. This is to allow TADL to discover all the possible paths during the searching process. These simulation results show that the TADL approach is effective in a stationary network or a network with a low to medium level of mobility. However, in a network with a high level of node mobility, links break very frequently. When a link break, packet delivery will fail, and different from packet dropping attacks, when a packet is not delivered due to link breaks caused by node mobility, the sending node will return a Route REFor (RRER) packet to the source node.

For each RRER packet received, the source node restarts a new route discovery process, if the source node did not find alternative path(s), which results in more control packets. In addition, the trust-based route selection approach used in TADL does not help to find more reliable paths when the node mobility level is high. The estimated trust value for a path in a previous instance is likely to be invalid. In addition, using Area-S will lead to fewer routes being discovered and if these routes become invalid quickly and frequently, more discoveries are required, leading to an even higher level of routing overhead than LAR. Therefore, one of the lessons learnt from these investigations is that the dynamic adjustment measure used in TADL is not effective or may be counterproductive in a highly mobile network. In LAR, however, the increase in the number of control packets as the result of broken links is not as significant, as TADL, but in TADL, more RREQ will be passing the intermediate nodes for each search leading to a higher level of routing overhead.

We have also examined the packet delivery ratios with varying Levels of traffic loads, attacker ratios and network node mobility levels. From Figures 3(d), 3(e) and 3(f), it can be seen that, TADL generally performs better than LAR under almost all network conditions. As shown in Figure 3(d), when the pause time is 300s and the attacker ratio is 0%, the delivery ratios by TADL maintains largely constant as the network load increases. However, in the case of LAR, the PDR ratios decreases as the network load increases. This also means that the routing overhead reduc-
This may be due to the fact that TADL introduces less traffic overhead because of using search area resizing, and, in addition, by selecting and using more reliable paths, the chances of successful packet deliveries is higher, which, in turn, further reduce the level of control overheads. All these factors can reduce the chance of the network being congested. This could be confirmed by the average end-to-end packet delivery delay results shown in Figure 3(g), which shows that LAR experiences a higher delay than TADL. In other words, from these results, we can see that TADL is able to cope with a higher level of traffic load.

Intuitively, attacker ratios must have negative impacts on PDR. That is, the higher the attacker ratio, the lower the PDR values should be. However, this trend is not shown up in Figure 3(e), rather for both protocols, as attacker ratios increase, there is no much change in PDR values. This can be explained as follows. There are two factors which lead to PDR drops: one is packet dropping attacks and the other is network congestions. Packet dropping attacks cause packet loss thus leading to PDR drops. At the same time, these attacks reduce the traffic load in the network reducing the level of PDR drop caused by congestions. In other words, packet-dropping attacks can offset packet loss caused by network congestions.

This section studies the effect of varying network node mobility levels, traffic loads and attacker ratios on the average end-to-end packet delivery delay. Figures 3(g), 3(h) and 3(i) shows that TADL performs better than LAR under all the conditions with a greater difference in delay between TADL and LAR is 11.9ms at 900s pause time in Figure 3(i). This is a consequence of the following: (1) The reduced control overhead of TADL; In LAR, the high control overhead congests the network; the data packets take a longer time at the congested intermediate nodes to be served. (2) As TADL uses multiple paths, when a path is break there are alternative paths that will be used immediately without the need of starting a new route discovery process, which will reduce the delay. Whereas, in LAR the need of reconstructing paths results in longer delays. (3) TADL avoids imposing any additional computational and communication costs on intermediate nodes, that reduces the queuing and process time.

6. CONCLUSIONS AND FUTURE WORKS

This paper has presented the design and evaluation study of a novel protocol suite, TADL, which is aimed at finding a set of most trust-worthy paths for routing higher priority traffic towards a destination node with minimum overhead costs. TADL dynamically adjusts the size of the searching area in adaptation to the underlying network conditions, thus effectively reducing the routing overhead imposed in the network. TADL provides a much better performance than LAR and these improvements are achieved with very little routing overhead costs. The simulation results show that TADL is practical in a stationary network or a network with a low to medium level of mobility. However, the dynamic adjustment measure used in TADL is not effective in a highly mobile network, the results show a decrease in the protocol performance by PDR over LAR widens as the network load increases.
performance in a high mobile network. In the future work we will improve TADL route discovery protocol to adapt it to all mobility network levels. In addition, TADL does not differentiate behaviours by malicious attackers from those by overloaded nodes. However, in our future work we will revise the algorithm to address this weakness.

Acknowledgments

Helen Bakhsh gratefully acknowledges King Abdulaziz University (KAU) and Saudi Ministry of Higher Education for their financial support.

7. REFERENCES


