# Locally Multipath Adaptive Routing Protocol Resilient to Selfishness and Wormholes

F. Farhat, M. R. Pakravan, M. Salmasizadeh, M. R. Aref
Department of Electrical Engineering
Sharif University of Technology
Tehran, Iran
{farhat, pakravan, salmasi, aref}@ee.sharif.edu

## ABSTRACT

In recent years, many protocols have been proposed to secure the routing process in mobile ad-hoc networks (MANETs). However, there are still many security issues that have not been fully resolved by these algorithms. Selfishness of independent nodes locally degrades network routing efficiency. Also malicious nodes can easily corrupt the performance of the MANET by showing their Byzantine behaviors and so what is called a wormhole attack. One of the solutions to defeat the collaborating of malicious nodes is multipath routing algorithms that can withstand the failure of the primary path and provide alternate paths between the source and destination when such attacks happen.

In this paper, we propose locally multipath adaptive routing (LMAR) protocol, classified as a new reactive distance vector routing protocol for MANETs. LMAR can find an ad-hoc path without selfish nodes and wormholes using an exhaustive search algorithm in polynomial time. Also when the primary path fails, it discovers an alternative safe path if network graph remains connected after eliminating selfish/malicious nodes. The key feature of LMAR to seek safe route free of selfish and malicious nodes in polynomial time is its searching algorithm and flooding stage that its generated traffic is equi-loaded compared to single-path routing protocols but its security efficiency to bypass the attacks is much better than the other multi-path routing protocols. LMAR concept is introduced to provide the security feature known as availability and a simulator has been developed to analyze its behavior in complex network environments. Efficiency of the route discovery stage is analyzed and compared with the previous algorithms.

## Keywords

Ad-hoc Wireless Networks Security, Efficient Local-Multipath Adaptive Routing Protocol, Resilience to Selfishness and Wormholes, On-demand Distance Vector Routing

## 1. INTRODUCTION

Mobile Ad-hoc Networks (MANETs) are characterized by mobile hosts connected by wireless links communicating with each other without any infrastructure such as access point or base station. MANETs have many applications in a wide area of situations. MANET applications include military and emergency instances to establish an integrated network in an infrastructure-less location. Also MANETs are used for sensor networks, civilian applications, and ubiquitous computing. We could imagine most of the applications of infrastructure-based networks for MANETs with mobility feature.

Each ad-hoc node can only communicate with its neighbor nodes within its radio range. Each node to send a message to the node out of its radio range has to ask its neighbor nodes' help. So a multi-hop path between source and destination must be discovered. As the hosts of MANET move randomly, the topology of the MANET changes. MANET is usually established for impromptu decisions to achieve the specific goal. As mentioned we could conclude the main characteristics of MANET such as: wireless autonomous nodes, ad-hoc-based network, multi-hop routing protocols, mobility, and infrastructure-less.

Many routing protocols have been suggested to find multi-hop routes between some nodes. But selfish and malicious nodes can disrupt found paths easily. Routing protocol designers try to secure their schemes by using cryptography methods like confidentiality, integrity, authentication and non-repudiation. But these methods are not enough to establish connection. One of the main security services mentioned in network security is *availability* that is needed to assure establishing connection. Providing availability service is a challenging issue like DoS attack resiliency and cannot be achieved by using cryptography methods lonely.

We propose locally multipath adaptive routing (LMAR) protocol which finds safe ad-hoc routes free of selfish nodes and wormholes. Utilizing a reactive approach, proposed routing protocol uses an adaptive rerouting method to bypass selfish/malicious nodes, and also guarantees establishing a safe connection in polynomial-time if network graph remains connected after eliminating wormholes and selfish nodes.

In following sections, first we briefly review in section 2 some of key concepts such as on-demand routing, multipath routing, node selfishness and wormholes in MANETs. In section 3, we present the key concepts of our proposed approach; LMAR. We explain the network model, problem definition, attack scenario and solution strategy. We describe the structure of LMAR and its features in section 4. Some of its important features are analyzed in section 5. Simulation results are given in section 6.

# 2. FUNDAMENTAL CONCEPTS

## 2.1 On-demand Distance Vector Routing

Routing protocols in wireless networks are usually based on either distance vector or source routing and not link state routing algorithms. Routing algorithms in conventional networks require periodic broadcast of routing information by each router like Destination Sequenced Distance Vector (DSDV) [1] as a proactive (periodic) routing protocol. In distance vector routing, each router broadcasts to all of its neighboring routers the prospected distance to all other non-neighboring nodes. The neighboring routers then compute the shortest path to each node, like Ad-hoc On-demand Distance Vector Routing (AODV) [2] as a reactive (on-demand) routing protocol.

Another famous concept of routing is source routing, a technique where the source of the packet determines the complete sequence of the nodes of the established route. Each intermediate node attaches its address to the header of the packet like what happened to Dynamic Source Routing (DSR) [3]. In link-state routing, each router broadcasts to all of its neighboring routers its view of the status of each of its adjacent links; the neighboring nodes then compute the shortest distance to each node based upon the complete topology of the network like Optimized Link State Routing (OLSR) [4]. Some protocols like DSDV, called as proactive, are suitable for network environments where there are no node power consumption constraints. In MANETs where nodes have power constraints and the network has a much more dynamic nature, reactive or on-demand routing protocols are usually used. Routing table update procedure of on-demand routing protocol are different from the previous proactive approaches. Reactive or on-demand routing protocols establish and sustain paths from one source to its destination on an as needed basis. Compared to the proactive counterparts of the on-demand protocols, the routing discovery overhead is typically lower. On-demand route discovery reduced the cost of route maintenance towards periodic route discovery, because unused routes are not updated. On-demand route discovery is useful when the network traffic is sparse and related to a known subset of nodes. But data traffic suffers from the delay of route discovery and route maintenance procedure. Also these routing algorithms, in contrast with source routing, use a smaller header, because their packets doesn't convey the address list of passing routers except the address of source and destination.

Flooding process helps reactive protocols to discover routes. The traffic source floods an inquiry packet called Route Request Packet (RREQ) into the network to find a route to the desire destination. When an old route breaks because of a link breakage, flooding is also applied to maintain a new route instead of previous route. To have effective performance of reactive protocols flooding procedure should be controlled, because it consumes the most amounts of network resources like bandwidth and nodes' power.

All of above protocols are single path protocols, because they find only one shortest path in the specific topology of the network. If there are selfish nodes that do not cooperate for proper execution of data forwarding, or malicious nodes that collude to make wormholes, active data forwarding paths may be broken. So route discovery process must be performed again. Thus it necessitates flooding RREQs in the network again. In ad-hoc networks with dynamic topology, frequent route maintenance requests cause the high latency of new route discovery stage and affect the performance efficiency adversely.

## 2.2 Multipath Routing

Multipath on-demand routing protocols try to alleviate single path reactive routing protocols flaws by discovering multiple routes in a single route discovery procedure. Source nodes and intermediate nodes could find multi-paths sent by destination nodes. Route maintenance stage is needed only when all previous paths fail. So route maintenance latency and routing overheads decrease. By storing more available path to destination, higher performance efficiency could be achieved. Multipath routing in wired networks has been suggested to decrease data re-forwarding, control congestion, and deal with QoS issues. Other coercing advantages of applying multipath routing in MANETs are lower latency, higher fault tolerance, lower energy consumption, and higher robustness. Some on-demand multipath routing protocols have been suggested for ad hoc networks, including Shortest Multipath Routing Using Labeled Distances [5], Ad hoc On-demand Multipath Distance Vector (AOMDV) [6], Multipath Dynamic Source Routing (Multipath DSR) [7], Cooperative Packet Caching and Shortest Multipath (CHAMP) [8], Temporally Ordered Routing Algorithm (TORA) [9], Split Multipath Routing (SMR) [10], and Routing On-demand Acyclic Multipath (ROAM) [11]. SMR and MDSR are on the basis of dynamic source routing whereas AOMDV, ROAM, CHAMP and TORA are destination-sequenced distance vector routing based. Mentioned protocols establish multiple paths upon request, the data traffic could be distributed into multiple routes. But usually a single route is mainly employed and the other routes are applied only when the main route fails.

## 2.3 Selfishness and Wormholes

Noting the distributed and mostly uncontrolled nature of MANETs, there are possibilities of node misbehavior that can have serious impact on network operation. Active attacks and passive attacks essentially threaten mobile ad-hoc networks (MANETs). In an active attack, the malicious routers have to consume some power to execute the assault, but passive attacks are principally due to lack of cooperation with the purpose of saving energy selfishly. Nodes that perform active attacks with the aim of damaging other nodes by causing network recess are considered to be malicious while nodes that execute passive attacks with the aim of saving battery life for their own communications are considered to be selfish.

A selfish node that wants to save battery life for its own communication, may not take part in any communications known as Selfish Type-I or only takes part in the route discovery process to update its routing table, and selfishly doesn't participate in the data forwarding process known as Selfish Type-II [22]. Selfish nodes can endanger the correct network operation by simply not cooperating in the packet forwarding process (this attack is also known as the black hole attack). Single path ad hoc routing protocols can not cope with the selfishness problem when a selfish node exists in the active shortest path. The selfish node doesn't forward the data traffic, so another route discovery process must be implemented to get a new route with possibly cooperative nodes. Accordingly the network performance severely degrades.

The effect of node selfishness on the network throughput and end-to-end delay has been simulated by Michiardi et al [12] when the dynamic source routing protocol is employed. Traditional network security services consist of authentication and access control cannot handle selfishness as well as wormhole attack. Secure cooperation enforcement schemes such as Nuglets as a virtual currency [13], CONFIDANT [14], CORE [15], or Token-based cooperation enforcement [16] seem to offer reasonable solution. In a collaboration enforcement security scheme, router misbehaviour could be caught by the collaboration between numerous nodes supposing that most of routers do network tasks normally. We show that our multipath routing protocol is flexible towards selfishness, because it could bypass selfish nodes and choose cooperative clients.

A further shrewd type of active attack is the establishment of a virtual private connection or tunnel (called as wormhole [17]) between two colluding malicious routers in MANET topology (see Figure 1). This feat allows the wormhole nodes to short-circuit two points of network, so the normal flow of routing messages creates a virtual vertex cut in the network that is controlled by the two colluding attackers. Our protocol could adaptively change paths containing working links instead of wormholes. Malicious nodes can disrupt the correct functioning of a routing protocol or degrade the performance of data forwarding process by discarding attracted traffic. This new type of attack goes under the name of wormhole attack.

To defend against the destructive wormhole attack various works have been done [18,19]. To restrict the travelling space of any message, time-based solution has been proposed in [18]. Time-based mechanisms rely on strict synchronization between the network routers, thus they need to apply some extra equipments like GPS. A collaborative routing protocol through routers share directional messages assuming that routers are supplied with directional antennas is suggested in [19] to protect ad-hoc network against wormhole attacks.

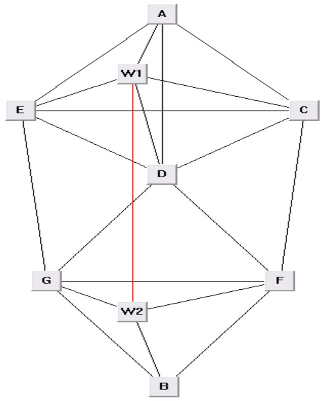

**Figure 1: Wormhole Link between W1 and W2**

Khabbazian et al. [23] have been suggested a single-path protocol against wormhole attack. In section IV.C they have been applied per packet*connection process run on all nodes to detect wormholes. In their attack scenario, only 4 consecutive nodes (F-M1-M2-E) have been provided that M1,M2 nodes are malicious and M1-M2 link is wormhole. If the nodes S-F-M1-M2-E-D construct the route from source (S) to destination (D), malicious behavior of M1-M2 affects on nodes F-E to choose as malicious, because they cannot provide the node S with a signed destination ACK or a signed RERR/FR packet too.

Our proposed protocol adaptively process per packet and changes the paths between the nodes to avoid wormholes, and it doesn't need to use any extra supplies. Detecting the wormhole attack has been suggested in [20] by a graph theoretic approach and [21] by a statistical approach.

## 3. OVERVIEW OF THE APPROACH

In this section we explain our assumptions about the network model and its security attributes. We then clarify the problem that endangers any mobile ad-hoc network. Thereafter we explain our solution strategy.

### 3.1 Network Model

Our proposed algorithm is implemented in the network layer. It is assumed that neighbor discovery phase has been carried out by an appropriate layer-2 or layer-3 processes. We also assume that an acknowledgement (ACK) reception process will execute in the transport layer or network layer that would indicate correct reception of source packets. When the source does not receive any ACK, it tries another route until it can establish a safe path. Our assumed network topology is a random graph model network, that ad-hoc nodes is randomly placed in their positions. Also it is assumed the nodes characteristics are the same.

### 3.2 Attack Scenario Model

Passive and active attacks on availability of safe connections are considered in this paper. Other threats against security services like confidentiality, integrity or non-repudiation by eavesdropping or forging packets have not been included, because some of them could be provided by using cryptographic methods.

Two kinds of selfish nodes including Type-I and Type-II could be available among the nodes. Behavior of a selfish/malicious node is independent from its neighbor nodes' behavior. In our attack scenario selfishness or wormholeness could happen occasionally by all the nodes so at the time T all nodes divided into ad-hoc and selfish/malicious nodes. Trying to establish selfish/malicious-free connections, ad-hoc nodes by using LMAR change their directions from time-variant selfish nodes at the time T+Δ.

Cooperation enforcement schemes need supervising by trusted third party. Third party is like a central bank that gives reputation to its branches. Without any third party, malicious nodes could cheat in reputations. They could generate fake currencies or shows the value of other nodes' reputation lower or higher. So we are not interested in cooperative schemes.

### 3.3 Problem Definition

As mentioned in Section 2.3, selfishness and wormholes threaten wireless ad-hoc networks. Selfish and malicious nodes participate in route discovery stage properly to update their routing table, but as soon as data forwarding stage begins, they discard data packets ungenerously! We want to suggest a routing scheme to solve these problems by ignoring routes crossing selfish nodes or wormholes. Mobility and its effect on the algorithm have not been widely discussed.

As shown in Figure 1, it is assumed that we want to establish a connection between source (A) and destination (B). Routes such AW1W2B (optimal), AEW1W2B, ACW1W2B, ADW1W2B, AW1W2GB, or AW1W2FB could be best choices for optimal and sub-optimal paths, but non of them would work when the wormhole thread runs, because all the high ranked paths include wormhole link between W1 and W2 (they discard all traffic containing data packets).

### 3.4 Solution Strategy

Our goal is to establish a connection between the source and the destination. To counter the effects of wormhole attacks and selfishness, LMAR algorithm creates at least one alternative working path that is established by active cooperators from source to destination. LMAR has been designed such that it could explore all available paths from source to destination in the graph of network topology to get a working route. Theoretically if the induced sub-graph of the network topology is connected by removing selfish and malicious nodes, LMAR can discover a locally efficient path from source to destination.

Referring to Figure 1, we have to get some reliable links to set up a desired route from A to B. Paths like ADGB, ADFB, AEGB, or ACFB are possible candidates. A flexible routing protocol must determine such paths as alternative routes to bypass the threats of wormholes or the selfishness of other nodes.

## 4. LMAR Protocol Structure

The design of LMAR is based on on-demand distance vector and multipath routing mentioned in sections 2.1 and 2.2. The protocol phases are divided into 3 parts: Route Discovery and Maintenance, and Data Forwarding.

### 4.1 Route Discovery and Maintenance

Every router has a clock measuring the number of ticks past first tick (when the router is on). As a source wants to transmit data to a destination and it doesn't know any active path to destination. Source node generates a proper RREQ packet to destination. Then it initiates a timer in transport layer to measure the period (here the number of ticks) between sending a RREQ and receiving its route reply (RREP). Subsequently it broadcasts the RREQ for destination.

The RREQ packets are a tuple with {Request ID (ReqID), Source Address (SrcAdd), Destination Address (DestAdd), Sequence Number (SeqNum), and Expiration Ticks (ExpTicks)}. *ReqID*, *SrcAdd*, and *DestAdd* are random unique identifiers (like IP Addresses or Nonce numbers) for RREQ, source, and destination respectively. As used in AODV, Sequence numbers play a key role in ensuring loop freedom.

An incremental sequence number is maintained by every node. Also the highest perceived sequence numbers for each destination node is kept in the routing table. The higher SeqNum shows a new route, so it is one of the parameter used to update routing table. *Tick* is the unit of delay cost in our implementation. Delay cost is defined for links as transmission delay and for routers as processing delay. While RREQ passes a hop, *ExpTicks* showing the lifetime of the RREQ is monotonically decreased. This technique is applied for flooding limitation.

Every intermediate node receiving RREQ checks the destination address. If the destination address is different from its address, it will update its routing table with respect to source address. Note that each RREQ (RREP) arriving at a node during route discovery potentially defines an alternate path back to the source (destination). All routes to the specific node in routing table are always less than the number of neighbors.

Each node can receive copies of RREQ from its neighbors, and update its routing table with respect to that destination. As we mentioned neighbor discovery phase has been carried out by an appropriate data-link or network layer processes attaching neighbor's identity as the uploader of the packet in the header of the frame.

Here delay cost instead of hop-count is applied for shortest path gauge. The sequence of paths is sorted with respect to the sequence number then the reception tick (*RecTick*), i.e. first the freshness of sequence number is checked and the path with the fresh SeqNum would be set as the first path. If the SeqNum is not changed, the *RecTick* is considered, and the path with lower *RecTick* would be set as the first path (The repeated neighbor is overwritten in the path list). The abstracted format of routing table simply is depicted below in Table 1.

Seeing Table 1, you could find that *Destination Address* is the RREQ or RREP creator address. The address of the RREQ or RREP owner is received by the owner of this routing table. In each table the *Entry Index* specifies the current active entry used for sending data towards the destination, and the default value of *Entry Index* is 1 that points to the first entry.

**Table 1: Abstracted Routing Table in LMAR**

| Destination Address | |
|---|---|
| Entry Index (Default=1) | |
| Index | The List of Paths (Entries) |
| 1 | {Neighbor1, SeqNum1, RecTick1} |
| 2 | {Neighbor2, SeqNum2, RecTick2} |
| 3 | {Neighbor3, SeqNum3, RecTick3} |
| | And so on |

After updating the routing table with respect to the source, intermediate node searches for any route to the intended destination of RREQ packet. If the number of paths to destination is equal with the number of neighbors, it will forward the RREQ to the best route with minimum delay cost; otherwise it will broadcast the RREQ to all neighbors. Each router broadcasts the RREQ only once like AODV protocol, but it may receive and process the copies of that to update routing table several times (at most the number of its neighbors).

When the first RREQ reaches the destination, it will be processed and the destination will generate RREP packet in response to RREQ packet. RREP packet structure is same as RREQ packet structure. RREP will be broadcast to source again. RREP flooding is necessary to set or update distance vector to the destination in intermediate nodes' routing tables. Also the destination doesn't involve in timeout process, because it couldn't measure the period between sending RREP and getting data packets, due to the fact that data packets never flood in network, so limited number of selfish or malicious nodes could discard them. Intermediate nodes apply the same process with RREP packets like what they do with RREQ packets. They update their routing table with respect to the destination address. If all possible routes to source are available, they will choose the best one to source, otherwise they will broadcast RREP packet again.

When the source gets the RREP packet generated by the destination, it updates its routing table, checks the RREP must not be repeated. Then it measures the period between request and reply, so it could set the ACK timeout period (the number of ticks needed to resend data packets in alternative route). Afterwards the best neighbor (first entry of routing table) as first hop of the shortest path is selected to send the desired traffic towards the destination.

When two nodes exit from the coverage range of each other, the available link between them is broken. So each node deletes its previous neighbor address from its routing table. Also when a node reaches the coverage range of another node, they have to set or update their reachable destinations by exchanging their Entry[Index][1] of their routing tables. Observe that these operations are performed in data link layer. No update packet is broadcast or multicast to other stable neighbors, only moving and separated nodes updates their tables.

## 4.2 Data Forwarding

As mentioned before, when the source gets a reply from the destination, it initiates data forwarding via the neighbor specified by the Entry[Index] of its routing tables. Also each intermediate node sets the reverse path index back to

[1] The Index$^{th}$ index of the routing table

the source in its routing table. Intermediate node gets the next hop of the best path to the destination, and forwards the data packet to its neighbor. If the data packet eventually arrives at the destination properly, the destination unicasts the ACK packet periodically after receiving multiple fragments of data packets. Sending an ACK from destination to source is a key process by which the presence of wormhole or selfish nodes in the path can be detected. This can be a part of network layer process or transport layer (e.g. using TCP ACK).

When ACK process reports an unestablished connection between source and destination which can be a sign of the presence of a wormhole or a selfish node in the path, a new route should be used. Therefore, the source chooses the next index of its routing table entries to establish a new route to destination. If all of entries are selected at least once, it means that the source has searched its table at least once (one round search reaches so the entry index sets to 1). Since then the source sets the *Change Path* flag in the header of next resent data packets to warn intermediate nodes for choosing alternative route to the destination. The interveners, which do not search all of its entries, reset the flag for subsequent nodes not to change their index, but if any intermediate node searched all of its routing table entries, it would keep the flag set adaptively for the next hops to change their index, and would forward the data packet to next neighbor. Observe that with this algorithm all of available paths between the source and the destination will be checked one by one in order to get a possibly safe route for data forwarding.

## 4.3 Main Features of LMAR

The proposed protocol has the following main features:

1. Our solution is absolutely compatible with ad-hoc nature in contrast to cooperation enforcement schemes based on trusted third party or a prior trust among nodes.
2. LMAR is a reactive routing protocol that finds routes on-demand to the destination, so overhead traffic of the protocol is lower than the most of multipath routing protocols.
3. It is designed to provide the security service known as availability. Most of other security services could be provided by using cryptographic methods like encryption, digital signature and hash function. A simulation environment has been developed to analyze its behavior in complex network environments.
4. Multi-level flooding in route discovery phase leads to setup routing tables of intermediate nodes related to the source of flood. Also generated traffic of LMAR's flooding is equi-loaded compared to single-path routing protocol.
5. LMAR locally detects multipath to the destination. Found paths by LMAR are locally efficient with the optimization parameter like transmission delay. Globally optimization consumes resource of the network severely.
6. Exhaustive search algorithm to find any connection is a non-polynomial time problem subject to the number of nodes in each level of flooding. But searching algorithm is designed to try the paths pseudo-randomly among the neighbors and routing table indexes are changing adaptively. So the probability of path finding increases but the cost of routes is locally not globally efficient.
7. LMAR adaptively changes its found route due to the behavior of intermediate nodes. Nodes could behave freely in ad-hoc, selfish or wormhole mode. LMAR adapts itself by social behavior of the nodes of the network.
8. It is properly robust to both Selfish type I and II [22]. It's also resilient to wormholes and malicious nodes while establishing connection. To address these issues most papers have not been completely successful.
9. LMAR does not need GPS, third party, watchdog, and time-synchronization infrastructures.
10. Rumor spreading is avoided by using first hand information. Comparing to similar protocols power loss reduces.
11. Our proposed protocol has auto-punishment strategy when an intermediate node changes routing table index from selfish/malicious node. As a game if punishment loss is higher than selfishness interest, selfish nodes encourage or enforce to cooperate.
12. Intermediate nodes do their routine activities per packet globally not per packet*connection. ACK process is end to end not hop by hop. So process time, memory and energy save for all the nodes but the latency of connection establishment increases.
13. The disadvantage of our protocol is that the end time of searching is unknown. LMAR could not get the final attempt of searching for any path between source and destination, if induced network graph is not connected. In fact there would be a trade-off between this disadvantage and the randomness of feature#6.

# 5. ANALYSIS

## 5.1 Resilience to Selfishness and Wormholes

LMAR protocol does a deep search on all of available routes between the source and the destination. So if the induced sub-graph of network topology containing the source and the destination without selfish and malicious nodes is connected, LMAR will find a valid path adaptively by an exhaustive search algorithm mentioned in section 4.2. The algorithm converges if and only if the induced sub-graph of ad-hoc nodes (containing source and destination) is connected, so theoretically there is a path (P) between source and destination that all nodes of the path (P) are ad-hoc not selfish or malicious. Single-path routing protocols like AODV and DSR could find only one path to some destination, so selfishness as a passive attack or wormhole attack (as an active attack) can disrupt the route between source and destination. Also previous multipath routing protocols like SMR [10] and AOMDV [6] could find a small finite set of paths that are link-disjoint or node-disjoint, it means that they do not search for all possible paths, so at the presence of wormhole attack or selfishness they couldn't provide availability service very well. As a result LMAR provides the availability security service correctly. In fact LMAR bypasses the selfishness of lazy nodes well. Wormholes in route discovery phase attract most paths toward themselves, so most of paths to some destinations nearer to wormhole are vulnerable, but LMAR changes the priority of their routing table due to the misbehaving of the wormholes. As mentioned in section 4.2, by not receiving ACK packet source knows to change the current path and if all paths doesn't work, source inform intermediates by setting Change Path flag, so the algorithm adaptively change the direction of the path.

## 5.2 Flooding Performance Comparison

LMAR protocol uses flooding twice, one for RREQ and one for RREP. Note that if each pair of nodes wants to communicate with each other (there are $(^{N}{}_{2})$ connections in a network with N nodes), we need to have only N-times flooding, because each node would only need to flood once in order to set its entry in the routing table of other nodes. So when the number of established connections increases, the number of essential flooding processes saturates to the number of network nodes. Furthermore the flooding process of LMAR is similar to AODV, because the policy for discarding repeated RREQ (and RREP for LMAR only) is the same. LMAR and AODV protocol broadcast each RREQ once, and LMAR protocol only updates routing table by other incoming copies of RREQ not rebroadcast, so the flooding cost of LMAR protocol per connection is 2*(AODV Flooding) and when the number of end to end connections increase, the flooding cost of LMAR reaches that of AODV, because RREQ or RREP packet of each node could be flooded in network once. Most of multipath routing protocols make an attenuated policy to discard repeated RREQs, e.g. if the repetitiveness criterion of RREQs is the similarity of *Request ID* and Incoming node simultaneously, more RREQs will be flooded in network, but if the repetitiveness policy only depends on the similarity of the *Request ID* like LMAR, RREQs coming from different nodes with similar *Request ID* will not be flooded anymore. The flooding process increases severely, when the number of non-repeated RREQ per connection flooding the network augments with respect to the number of incoming link (or the number of neighbors) or the number of disjoint paths. Hence the flooding performance of LMAR is better than some similar multipath protocols. Whereas LMAR uses on-demand distance vector routing, so the overhead of packets is relatively lower than other source routing protocols.

## 5.3 Route Optimality

If there is no selfish and malicious node, the route selected by LMAR is the shortest path. In the presence of selfishness or wormholes, discovered paths by LMAR are locally (not globally) efficient, because the globally efficient routes become passive by selfish or malicious nodes not cooperating with other nodes present in the path. When LMAR choose an alternative route, there is a node in selected route that its first entry has been assigned. So the selected path can be locally efficient but not globally optimal. The main disadvantage of our protocol is that the end time of searching is unknown. In fact LMAR could not find and stop searching for any route between the source and the destination, if induced network graph is not connected.

# 6. SIMULATION RESULTS

We developed a simulator on a PC under Windows XP operating system with Visual C++ programming to implement LMAR protocol. The simulation speed is higher than similar simulators like NS-2 or Glomosim, because we only simulate main parameters of our model and some unused codes is not executed periodically. Also our simulator has visual effects to show the topology changes visually. Furthermore simulator developing is cumbersome but the simulator code can be easily debugged in Visual C++ environment.

## 6.1 Selfishness Resilience

To show the resilience of LMAR to selfishness we set up a uniform-randomly generated network topology containing 50 similar nodes randomly placed over a 100*100 $m^2$ area (see Figure 2), so the network topology is a random graph. Source (node 0) and destination node (node 49) are marked by circles in the network. The radio range of all nodes is set to 30 pixels, so they can cover the nodes available in a circle with 30-pixel radius. Nodes mutually covering their radio ranges have been interconnected to each other by a black line. When the packets pass the links, the color of the links is changed. RREQs color the links darker than RREPs do. All nodes' transmission range is 10m.

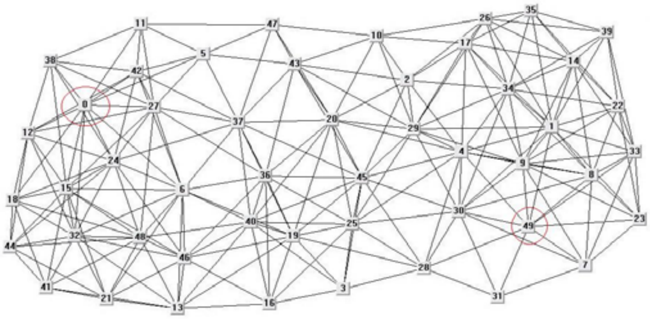

**Figure 2: A sample network topology**

When the simulation is performed, RREQs are broadcasted in network (the links go darker), also RREPs are broadcasted (the links go brighter). As shown in Figure 3, in the middle of the simulation we have some dark and bright links together.

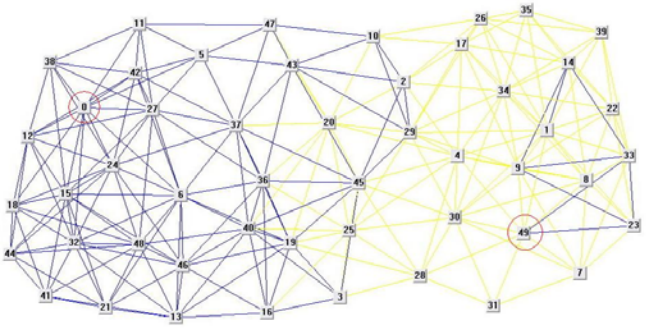

**Figure 3: the broadcasting of RREQs and RREPs**

Afterwards node 0 sends the traffic towards node 49 by choosing the shortest path in its routing table. The intermediate nodes send the traffic towards their first index entry, the best path. The purple line[1] in Figure 4 shows the forwarding path from the source to the destination (this line is overwritten by the return path of ACK packet). The ACK packet traverses the reverse path (the green line) back to the source, as the intermediate nodes set the reverse hop back to the source in their routing tables. So the green line of reverse path overcolor the purple line of forwarding path (we only showed the final reverse path in Figure 4).

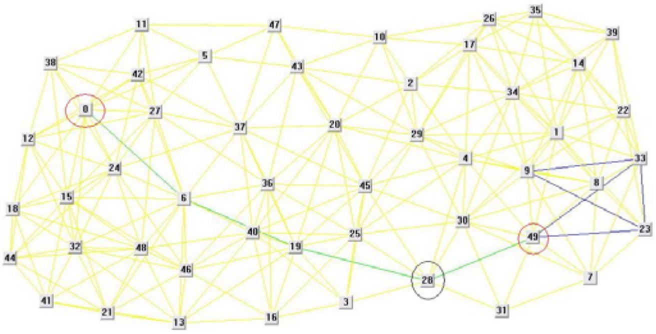

**Figure 4: Established connection with the shortest path**

As node 28 the neighbor of node 49 illustrated in Figure 4 with a black circle around, we change the characteristic of node 28 from *ad-hoc* to *selfish*, and run the simulation again. The details of RREQs and RREPs broadcasting are eliminated and the final result that is the alternative path to forward data is shown in Figure 5.

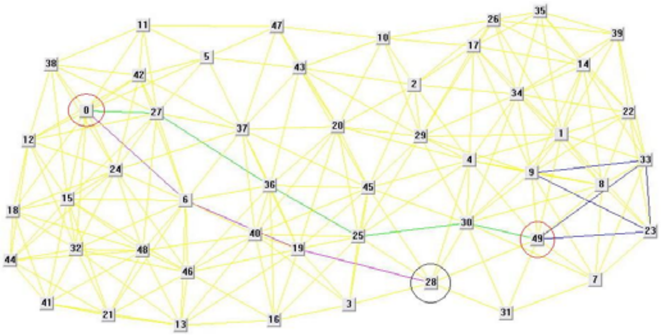

**Figure 5: LMAR resilience to one node selfishness**

Now we change the characteristics of nodes 30 another neighbor of node 49 as a selfish node. Our simulator showed that the locally efficient path would be the route containing the nodes of 0-42-37-20-4-49. So we put the node 4 as a selfish node too. As depicted in Figure 6, the number of timeouts or attempts to establish connection increases (purple line shows the tried paths, but only the green path sends the ACK back to the source).

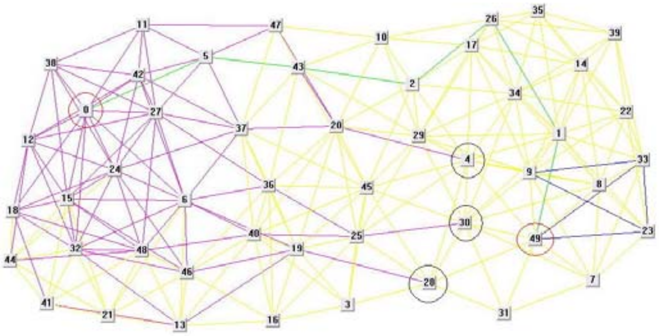

**Figure 6: LMAR resilient to neighbors' selfishness**

In spite of the fact that the number of tries to set up connection increases when the number of selfish nodes

[1] Note that the forwarding route is darker than the reverse path of ACK packet

grow up, every try could change the index entry of intermediate nodes away from selfish nodes (even malicious nodes) spontaneously as mentioned in section 4.2. The increment of tries is a good work for prospective end to end connections, because intermediate nodes have changed their index entry for next connection establishment attempts away from selfish nodes, so the number of tries decreases for the subsequent connections.

### 6.2 Wormhole Attack Resilience

As illustrated in Figure 7, to show the resilience of LMAR to wormholes we use previous random topology, and add a wormhole link (the red line) between node 24 (the neighbor of node 0) and node 30 (the neighbor of node 49). Obviously the shortest path between node 0 and node 49 would be the path containing the nodes of 0-24-30-49.

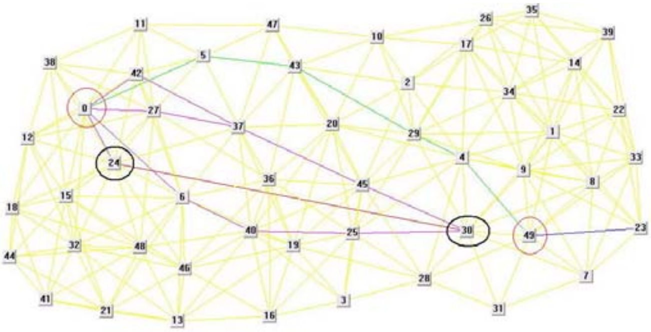

**Figure 7: LMAR resilience to wormhole attack**

As seen in Figure 7, node 0 as the source could find a locally efficient route to node 49 as the destination after 5 tries. Also it shows that the wormholes are more dangerous than selfishness, because they attract most of traffics towards themselves. Longer wormholes have a much more severe impact on the network performance and LMAR requires many more steps to converge. When the number of tries increases, the direction of intermediate nodes' index entry switches away from wormhole contributors similar to what happened when the number of selfish nodes increased in the example given before.

Our goal is connection establishment or providing availability services. The cost of our aim is the flooding of route discovery stage. So we developed our simulator to show the number of RREQ/RREP transmission. Also one of similar protocol called AOMDV is simulated and the results of our solution are compared to AOMDV below. There are 15 random connections in this scenario and the number of selfish nodes out of prospective connected nodes is incremented one by one.

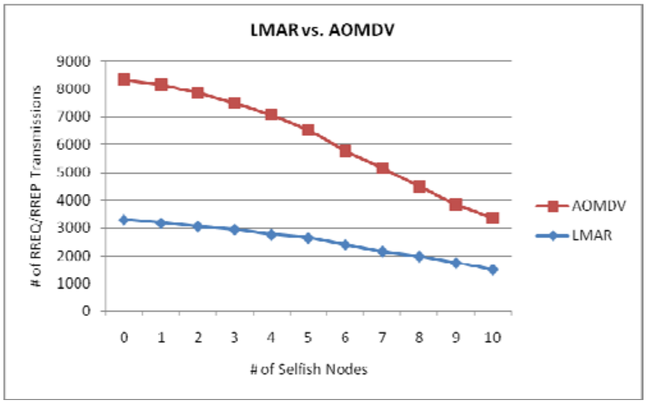

**Figure 8: Number of RREQ/RREP Transmissions Comparison**

As shown in Figure 8, the number of RREQ/RREP transmissions of LMAR is much better than a typical reactive multi-path routing protocol (AOMDV). As mentioned the overload traffic of flooding stage of LMAR is comparable with single-path routing protocol like AODV and in full loaded traffic they reach each other. By incrementing the number of selfish nodes (Type I) many packets drops and the throughput of AOMDV approaches to LMAR. After route discovery stage, LMAR as shown in previous figures can setup 15 expected connections, but AOMDV's found paths are not enough to establish some connections.

## 7. CONCLUSION

In this paper, we introduced an efficient local-multipath adaptive routing (LMAR) protocol, a routing algorithm to provide availability security service properly. The objective of LMAR design is to provide a mechanism by which network nodes can find alternative paths using a distributed mechanism. This is done using an exhaustive search algorithm. Therefore LMAR can bypass the selfishness of independent nodes that disrupt the data path. In addition LMAR could defeat the wormhole attach by finding alternate routes that bypass the wormholes.

## 8. ACKNOWLEDGMENT

This work is partially supported by Iran Telecommunication Research Center (ITRC) in Information Systems and Security Lab (ISSL), Electrical Engineering Department, Sharif University of Technology. Also we thank the Iranian NSF for support and establishing the cryptography chair.

## 9. REFERENCES

[1] C. E. Perkins and P. Bhagwat, Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers, Proc. of the ACM SIGCOMM, 1994.

[2] C. E. Perkins, E. M. Royer, and S. R. Das, Ad-hoc On-demand Distance Vector (AODV) routing, IETF MANET Group, Jan 2002.

[3] D. B. Johnson, D. A. Maltz, Y. C. Hu, and J. G. Jetcheva, The Dynamic Source Routing Protocol for Mobile Ad-hoc Networks, IETF MANET Group, Feb 2002.

[4] T. Clausen and P. Jacquet, RFC 3626: Optimized link state routing protocol (OLSR), October 2003.

[5] C. Balasubramanian and J.J. Garcia-Luna-Aceves, Shortest Multipath Routing Using Labeled Distances, Proc. Of 1st IEEE MASS, October 2004.

[6] M. K. Marina and S. R. Das, on-demand multipath distance vector routing in ad hoc networks, Proc. IEEE ICNP, 2001.

[7] A. Nasipuri, R. Castaneda, and S. R. Das, Performance of Multipath Routing for On-Demand Protocols in Mobile Ad Hoc Networks, ACM/Kluwer Mobile Networks and Applications (MONET), 2001.

[8] A. Valera, W. Seah, and S. V. Rao, Cooperative packet caching and shortest multipath routing in mobile ad hoc networks, Proc. IEEE INFOCOM, 2003.

[9] V. D. Park and M. S. Corson, A highly adaptive distributed routing algorithm for mobile wireless networks, Proc. IEEE INFOCOM, 1997.

[10] S. Lee and M. Gerla, Split Multipath routing with maximally disjoint paths in ad hoc networks, Proc. IEEE ICC, 2001.

[11] J. Raju and J. J. Garcia-Luna-Aceves, A new approach to on-demand loop-free multipath routing, Proc. IC3N, 1999.

[12] P. Michiardi, R. Molva, Simulation-based Analysis of Security Exposures in Mobile Ad Hoc Networks, Proc. European Wireless Conference, 2002.

[13] L. Buttyan, J.-P. Hubaux, Nuglets: a virtual currency to stimulate cooperation in self-organized ad hoc networks, Technical Report DSC/2001/001, 2001.

[14] S. Buchegger, J.-Y. Le Boudec, Nodes Bearing Grudges: Towards Routing Security, Fairness, and Robustness in Mobile Ad Hoc Networks, 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing.

[15] P. Michiardi, R. Molva, CORE: A Collaborative Reputation mechanism to enforce node cooperation in Mobile Ad Hoc Networks, IFIP - Communication and Multimedia Security Conference 2002.

[16] H. Yang, X. Meng, S. Lu, Self-Organized Network-Layer Security in Mobile Ad Hoc Networks.

[17] A. Perrig, Y-C Hu, D. B. Johnson, Wormhole Protection in Wireless Ad Hoc Networks, Technical Report TR01-384, Dep. Of Computer Science, Rice University.

[18] Y.-C. Hu, A. Perrig and D. B. Johnson, Packet Leashes: A Defense against Wormhole Attacks in Wireless Ad Hoc Networks, Proc. of INFOCOM 2003, April 2003.

[19] L. Hu and D. Evans, Using Directional Antennas to Prevent Wormhole Attacks, Network and Distributed System Security Symposium (NDSS), San Diego, Feb 2004.

[20] L. Lazos, R. Poovendran, C. Meadows, P. Syverson, L. W. Chang, Preventing Wormhole Attacks on Wireless Ad-hoc Networks: A Graph Theoretic Approach, IEEE Wireless Communications and Networking Conference, 2005.

[21] N. Song, L. Qian, X. Li, Wormhole Attacks Detection in Wireless Ad-hoc Networks: A Statistical Analysis Approach, Proc. of the 19th IEEE International Parallel and Distributed Processing Symposium, 2005.

[22] B. Wang, S. Soltani, J. Shapiro and P. Tan, “Local detection of selfish routing behavior in ad hoc networks”, Proceedings of 8th international symposium of parallel architectures, algorithms and networks, IEEE, 2005.

[23] M. Khabbazian, H. Mercier, V.K. Bhargava, "Wormhole Attack in Wireless Ad Hoc Networks: Analysis and Countermeasure", Global Telecommunications Conference, IEEE, 2006.

.