CHAPTER VI

6. REPUTATION BASED ON-DEMAND ROUTING PROTOCOL USING MOBILE AGENT

This chapter explains the core contribution of the thesis - the reputation based routing protocol using mobile agents.

6.1 MOBILE AGENT

This section explains the mobile agent in detail.

6.1.1 A Software Agent

As per the IBM’s definition given by Lange and Oshima (1998), an agent is a software object that:

- is situated within an execution environment;
- possesses the following mandatory properties:
  - Reactive - senses changes in the environment and acts according to those changes;
  - Autonomous - has control over its own actions;
  - Goal driven - is pro-active;
  - Temporally continuous - is continuously executing
- and may possess any of the following orthogonal properties:
  - Communicative - able to communicate with other agents;
  - Mobile - can travel from one host to another;
  - Learning - adapts in accordance with previous experience;
  - Believable - appears believable to the end-user.
6.1.2 Mobile Vs. Stationary Agents

Mobility is an orthogonal property of agents. That is, all agents are not necessarily required to be mobile. An agent can remain stationary and communicate with the surroundings by conventional means like remote procedure calls (RPC) and remote object invocation (RMI) etc. The agents that do not or cannot move are called stationary agents. On the other side, a mobile agent is not bound to the system where it begins execution. The mobile agent is free to travel among the hosts in the network. Once created in one execution environment, it can transport its state and code with it to another execution environment in the network, where it resumes execution.

6.1.3 Mobile Agents and Mobile Agent Environment

A mobile agent must contain all of the following models: an agent model, a life-cycle model, a computational model, a security model, a configuration model and finally a navigation model. A working definition of a mobile agent can is given by Jain, Anjum, and Umar (2000) as follows:

“A mobile agent consists of a self-contained piece of software that can migrate and execute on different machines in a dynamic networked environment, and that senses and (re) acts autonomously and proactively in this environment to realize a set of goals or tasks.”

The software environment in which the mobile agents exist is called mobile agent environment. Following is the definition of mobile agent environment referred by Kunal shah (2003):
“A mobile agent environment is a software system distributed over a network of heterogeneous computers. Its primary task is to provide an environment in which mobile agents can execute. It implements the majority of the models possessed by a mobile agent.”

The above definitions state the essence of a mobile agent and the environment in which it exists. It is given by Mahmoud (2001). The mobile agent environment is built on top of a host system. Mobile agents travel between mobile agent environments. They can communicate with each other either locally or remotely. Finally, a communication can also take place between a mobile agent and a host service.

6.1.5 Advantages of Mobile Agents

The mobile agents have several strengths. The following is the brief discussion of seven good reasons for using mobile agents again given by Lange and Oshima (1999):

- They reduce network load: The main motivation behind using mobile agents is to move the communication to the data rather than the data to the computations. Distributed systems often required multiple interactions to complete a task. But using mobile agent allows us to package a conversation and send it to a destination host. Thus all the interactions can now take place locally. The result is enormous reduction of network traffic. Similarly instead of transferring large amount of data from the remote host and then processing it at the receiving host, an agent send to the remote host can processed the data in its locality.

- They overcome network latency: Certain real-time systems require immediate action in response to the changes in their environment. But a central controller cannot respond immediately due to the network latency.
Here mobile agents can be a good solution as they can be dispatched from a central controller to act locally in the system and thus can respond immediately.

- They encapsulate protocols: Due to the continuous evolution of existing protocols in a distributed system, it is very cumbersome to upgrade protocol code property in each host. Result may be that protocols become a legacy problem. Mobile agents are able to move to remote hosts in order to establish “channels” based on proprietary protocols.

- They execute asynchronously and autonomously: This is the reason why mobile agents are so promising in wireless networks. Due to the fragile and expensive wireless network connections, a continuous open connection between a mobile device and a fixed network will not be always feasible. In this case the task of the mobile user can be embedded into mobile agents, which can then be dispatched into the fixed network and can operate asynchronously and autonomously to accomplish the task. At a later stage the mobile user can reconnect and collect the agent with the results.

- They adapt dynamically: Mobile agents are capable of sensing their execution environment and take decisions based on that dynamically.

- They are naturally heterogeneous: Mobile agents are generally independent of the computer and the transport layer and depend only on their execution environment. Hence they can perform efficiently in any type of heterogeneous networks.

- They are robust and fault-tolerant: The dynamic reactivity of mobile agents to unfavorable situations makes it easier to build robust and fault-tolerant distributed systems.
6.1.6 Disadvantages of mobile agents

The main drawback of mobile agents is the security risk involved in using mobile agents. A malicious node can disperse a mobile agent which can damage a host. For example, a virus can be disguised as a mobile agent and distributed in the network causing damage to the host machines that execute the agent.

6.2. THE MOBILE AGENT MODEL

The mobile agent used in this thesis has the following components:

1. Mobile agent identifier
2. Mobile code
3. Mobile agent stack

The identifier is used to uniquely identify the mobile agent from multiple copies. The mobile code is the software which performs some specific operations which is described later. The stack is used to store the address of the nodes visited and the reputation values of the corresponding nodes.

The mobile agent is designed to perform:

- Once initialized, moves to all the neighbouring nodes of the source node.

- On reaching each node, it checks whether the node is already visited by checking the stack.

- If it is already visited, the mobile agent dies there. Otherwise, the address of the visited node is added in the stack.
If the visiting node is the destination, the mobile agent searches the reputation value of the first node in the stack. This value is attached with corresponding node id.

The mobile agent starts to migrate from the destination to the source by using the addresses in the stack.

In the return path, at each node the mobile agent attaches the reputation value of the next node in the stack to the corresponding node id.

Finally the mobile agent has a route to the destination with the addresses of the nodes along with reputation values.

Mobile agent performs well in collecting and processing the information. Usage of mobile agent for collecting route information reduces the overhead in the network and also reduces the time taken to receive a Route Reply.

6.3 THE APPROACH

The approach in this thesis is divided in to two phases. One phase is Route discovery and another phase is data transfer. Route discovery phase is done in two parts. The route maintenance mechanism is same as DSR hence it is not explained here.

6.3.1 Route Discovery Phase

The first part of the route discovery mechanism is, when a node needs to discover a route, it broadcasts a mobile agent with a unique identifier and the destination address as parameters. It also has a stack (briefcase) for carrying the node ids and reputation values. A mobile agent when reaches a
neighbouring node will do the following. If the mobile agent has already visited the node it dies there. If the mobile agent recognizes the node’s address as the destination, it has reached the target. Otherwise the mobile agent appends the node’s address to a list of traversed hops and migrates to next node. Using this approach the mobile agent collects a list of addresses representing a possible path on its way towards the destination. Thus as soon as the mobile agent reaches the destination it can return the route to the source node using the list of visited nodes as reverse path. A mobile agent is a composition of computer software and data which is able to migrate from one computer to another autonomously and continue its execution on the destination computer.

The second part of the protocol is to attach the reputation values of each node in the route. Each node maintains a list of reputation values of their neighbours got by using monitoring mechanism. This value is updated for every 5 minutes (in our simulation). While returning from the destination to the source, the mobile agent follows the reversed list of addresses. In each node, the mobile agent searches the reputation value of the next node in the list and attaches the reputation value along with the address of that node. Finally the mobile agent reaches the source with the addresses of the nodes in the route and their corresponding reputation value. At the source there may be more than one route to the same destination which is returned by the copies of mobile agent. Now the source can decide the best path from the routes by comparing the reputation values of various routes. For example the two copies of the mobile agent will have the stack value in the format shown below.
One route:

<table>
<thead>
<tr>
<th>Node ID</th>
<th>Reputation value</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>0.6</td>
</tr>
<tr>
<td>7</td>
<td>0.8</td>
</tr>
</tbody>
</table>

Another route:

<table>
<thead>
<tr>
<th>Node ID</th>
<th>Reputation value</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.2</td>
</tr>
<tr>
<td>5</td>
<td>0.9</td>
</tr>
</tbody>
</table>

Figure 6.1 Sample routes collected by mobile agent for source node 1

From these routes the source will decide the best path by taking the average reputation value of the nodes in the path. For the first route in the example the average reputation value is \((0.6+0.8)/2 = 0.7\). For the second route; \((0.2+0.9)/2 = 0.55\). Since the average reputation value through route 1 is greater than route 2 the source selects route 1 as best path. Note that the source can also reject a route by checking the reputation value of individual nodes e.g. if it is below the threshold value say 0.5. Using this protocol it is possible to find a path with less number of misbehaving nodes, thus achieving a reliable communication.

6.3.2 Data Transfer Phase

When a node wants to transmit a packet, it finds the route using route discovery process. A comparatively trustworthy route is now ready for transmitting the data, but a node’s data will be forwarded by the neighbouring
node only if its reputation value is greater than the threshold value. Consider the following example network shown in Figure 6.2.

![Diagram showing a network with nodes 1, 2, 3, 4, 5, 6, 7, 8, 9 and a route from node 1 to node 9 through nodes 4 and 7]

**Fig 6.2 Example trustworthy route from node 1 to node 9 thru nodes 4 and 7**

Suppose node 1 finds a trustworthy route through nodes 4 and 7 to reach node 9. Node 1 transmits the packet to node 4. Node 4 can forward the packet of node 1 only if its reputation value is greater than 0.8, otherwise node 4 will drop the packets. Thus to get the service of the network, a node has to maintain a good reputation value by forwarding other nodes’ packets.

### 6.4 ROUTING ATTACKS

Even though the main concentration of the thesis is on selfish nodes, some types of malicious behaviours can also be controlled by this approach. Routing attacks like rushing attack, wormhole attack and gray-hole attack can also be mitigated by using the reputation mechanism. This is explained in more detail, in the next sub sections.
6.4.1 Rushing Attack

The rushing attack as explained by YihChun Hu et al., (2003) is a malicious attack that acts as an effective denial of service attack against all currently proposed on-demand ad-hoc network routing protocols (e.g. DSR, AODV etc.) including protocols that were designed to be secure (e.g. Ariadane, ARAN etc.) In on-demand routing protocol, the route discovery is a mechanism by which a source node obtains a route to destination node when wish to send a packet to it. Normally, the source node obtains a suitable source route by searching its route cache of routes previously learned, but if no route is found in its cache, it will initiate the route discovery by flooding the Route Request (RREQ) to the network. To limit the overhead of this flood, each node typically forwards only one Route Request (RREQ) originating from any route discovery. In particular, existing on-demand routing protocols, such as AODV, DSR, LAR, Ariadne, SAODV, ARAN, AODV secured with SUCV and SRP, only forward the request that arrives first from each route discovery. In the rushing attack, the attacker exploits this property of the operation of route discovery. The aim of the malicious node is to rush the Route Request to neighbours of the destination node so that it first affects the route discovery mechanism and later it can affect the route maintenance mechanism, routing etc.,

6.4.2 Rushing attack in DSR

In DSR, during the route discovery, a malicious intermediate node (in a network containing more than two hops) without verifying the Route Request in its route cache, forwards the Route Request whose aim is to directly rush the Route Request towards the neighbour of the destination node. During the Route Reply (RREP) of the destination, the malicious node takes a place as an intermediate node in the route accumulation of the route discovery.
As a result, a route containing malicious node is created between sources to destination in the network. That is, when a neighbour of the target receives the rushed request from the attacker, it forwards that request. It will not forward any further requests from this route discovery. When the legitimate requests arrive later at these nodes, they will be discarded.

**Fig 6.3 Example for rushing attack, nodes 7 and 8 are malicious**

In figure 6.3, the node 1 initiates a Route Discovery for the target node 9. If the Route Requests for this Discovery forwarded by the attacker are the first to reach each neighbour of the target (shown in gray in the figure 6.3), then any route discovered by this Route Discovery will include a hop through the attacker. That is, when a neighbour of the target receives the rushed request from the attacker, it forwards that request, and will not forward any further requests from this Route Discovery. When non-attacking requests arrive later at these nodes, they will discard those legitimate requests. As a result, the initiator will be unable to discover any usable routes (i.e., routes that do not include the attacker) containing at least two hops (three nodes).

In general terms, an attacker that can forward Route Requests more quickly than legitimate nodes can do so. It can increase the probability
that routes that include the attacker will be discovered rather than other valid routes. Whereas the discussion above has used the case of nodes that forward only the first Route Request from any Route Discovery, the rushing attack can also be used against any protocol that predictably forwards any particular request for each Route Discovery. A rushing attacker need not have access to vast resources.

On-demand routing protocols delay Route Request forwarding in two ways. First, Medium Access Control (MAC) protocols generally impose delays between when the packet is handed to the network interface for transmission and when the packet is actually transmitted. In a MAC using time division, for example, a node must wait until its time slot to transmit, whereas in a MAC using carrier-sense multiple access, a node generally performs some type of back-off to avoid collisions; protocols like IEEE 802.11 also impose an inter-frame spacing time before transmission actually begins. Second, even if the MAC layer does not specify a delay, on-demand protocols generally specify a delay between receiving a request and forwarding it, in order to avoid collisions of the request packets. In particular, because request packets are broadcast and collision detection for broadcast packets is difficult, routing protocols often impose a randomized delay in request forwarding. An attacker ignoring delays at either the MAC or routing layers will generally be preferred to similarly situated non-attacking nodes. One way to thwart an attacker that rushes in this way is to remove these delays at both the MAC and routing layers, but this approach does not work against all types of rushing attackers and is not general. For example, in a dense network using a CSMA MAC layer, if a node $A$ initiates a Route Discovery, and $B$ is two hops away from $A$, and $C$ and $D$ are neighbours of both $A$ and $B$, then $B$ will likely not receive the Route Request due to a collision between requests forwarded by $C$ and $D$. In a dense network, such collisions may often prevent the discovery of any
nontrivial routes (routes longer than a direct link), which is even more severe than the rushing attack, which prevents the discovery of routes longer than two hops.

Another way that a relatively weak attacker can obtain an advantage in forwarding speed is to keep the network interface transmission queues of nearby nodes full. For example, if each node processes the packets it receives in order, and an inefficient request authentication mechanism is used, the attacker can keep other nodes busy authenticating requests containing bogus authentication, thus slowing their ability to forward legitimate requests. Protocols employing public key techniques are particularly susceptible to these attacks, since they require substantial computation to validate each received request. A relatively weak attacker can also achieve faster transit of its request packets by transmitting them at a higher wireless transmission power level, thus reducing the number of nodes that must forward that request to arrive at the target. Since packet transit time at each hop is dominated by the processing time at the forwarding node, reducing the path to the target by just one hop is likely to provide a significant latency advantage, thus strengthening the attackers’ position. A more powerful rushing attacker may employ a wormhole to rush packets. In this case, the attacker simply forwards all control packets (but not data packets) received at one node (the attacker) to another node in the network (e.g., a second attacker). This forms a tunnel in the network, where packets reaching one end of the tunnel are broadcast out the other end. If the tunnel provides significantly faster transit than legitimate forwarders, nodes near one end of the tunnel generally will be unable to discover working routes to the other end of the tunnel, since it will generally discover routes through the tunnel. In general, a wired tunnel (in which the two attackers have a wired connection between themselves) will provide faster
transit than native wireless (multi-hop) forwarding, since node processing delay in forwarding is much longer than the propagation time.

The rushing attack applies to all proposed on-demand protocols because such protocols must limit the number of packets that any node will transmit in response to a single Route Discovery. Currently proposed protocols choose to forward at most one request for each Discovery; any protocol that allows an attacker to predict which Route Request(s) will be chosen for forwarding at each hop will be vulnerable to some variant of the rushing attack.

6.4.3 Avoiding Rushing attacks using mobile agents

The approach used in the thesis avoids rushing attacks by using the mobile agents. Instead of Route Requests, the mobile agents are doing the process of route discovery. There is no chance of malicious nodes replying to the Route Requests with rush. Hence no problem of adding any unwanted nodes inside the route list. An important problem of on-demand routing protocols are thus solved simply by using the mobile agents.

6.4.4 Wormhole attack

A wormhole attack is composed of two attackers and a wormhole tunnel. To establish a wormhole attack, attackers create a direct link, referred to as a wormhole tunnel, between them. Wormhole tunnels can be established by means of a wired link, a high quality wireless out-of-band link or a logical link via packet encapsulation. After building a wormhole tunnel, one attacker receives and copies packets from its neighbours, and forwards them to the other colluding attacker through the wormhole tunnel.
This latter node receives these tunneled packets and replays them into the network in its vicinity. In a wormhole attack using wired links or a high quality wireless out-of-band links, attackers are directly linked to each other, so they can communicate swiftly. However, they need special hardware to support such communication. On the other hand, a wormhole using packet encapsulation is relatively much slower, but it can be launched easily since it does not need any special hardware or special routing protocols.

(Mahajan et al., 2008) proposed some proposals to detect wormhole attacks like:

1) The abrupt decrease in the path lengths can be used as a possible symptom of the wormhole attack.

2) With the available advertised path information, if the end-to-end path delay for a path cannot be explained by the sum of hop delays of the hops present on its advertised path, existence of wormhole can be suspected. The advertised routes are much shorter than the actual routes which go through the wormhole tunnel.

3) Some of the paths may not follow the advertised false link, yet they may use some nodes involved in the wormhole attack. This will lead to an increase in hop delay due to wormhole traffic and subsequently an increase in end-to-end delay on the path. An abrupt increase in the end-to-end delay and the hop queuing delay values that cannot be explained by the traffic supposedly flowing through these nodes can lead us to suspect the presence of wormhole.

An attacker records packets at one location in the network and tunnels them to another location. Routing can be disrupted when routing control messages are tunneled. This tunnel between two colluding attackers is
referred as a wormhole. Wormhole attacks are severe threats to MANET routing protocols. For example, when a wormhole attack is used against an on-demand routing protocol such as DSR or AODV, the attack could prevent the discovery of any routes other than through the wormhole.

![Fig.6.4 Example for a wormhole attack M1 and M2 malicious nodes](image)

In any ad-hoc network, a wormhole can be created through the following three ways:

- Tunneling of packets above the network layer
- Long-range tunnel using high power transmitters
- Tunnel creation via external wired infrastructure

In the first type of wormhole, all packets which are received by a malicious node are duly modified, encapsulated in a higher layer protocol and dispatched to the colluding node using the services of the network nodes.
These encapsulated packets traverse the network in the regular manner until they reach the collaborating node. The recipient malicious node, extracts the original packet, makes the requisite modifications and sends them to the intended destination.

Figure 6.4 shows an example for a wormhole attack of first type explained above. In the second and third type of wormholes, the packets are modified and encapsulated in a similar manner. However, instead of being dispatched through the network nodes, they are sent using a point to-point specialized link between the colluding nodes.

6.4.5 Avoiding wormhole attack using mobile agent

In the approach which is used in this thesis, a mobile agent is broadcasted instead of Route Requests. A mobile agent is an autonomic entity which moves from one hop to another hop independently and collects the route information. So no single or collaborative routing attacks like wormhole attack can affect our mechanism. The mobile agent reaches a mobile node, updates its source list, and searches the reputation table in the return path and leaves.

6.4.6 Black hole attack

Bing Wu et al., (2006) define two properties for black hole attacks. First, the node exploits the mobile ad-hoc routing protocol, such as DSR, to advertise itself as having a valid route to a destination node, even though the route is spurious, with the intention of intercepting packets. Second, the attacker consumes the intercepted packets without any forwarding. However, the attacker runs the risk that neighbouring nodes will monitor and expose the ongoing attacks. There is a more subtle form of these attacks when an attacker selectively forwards packets. An attacker suppresses
or modifies packets originating from some nodes, while leaving the data from the other nodes unaffected, which limits the suspicion of its wrongdoing.

In black hole attack, the malicious node waits for the neighbours to initiate a RREQ packet. As the node receives the RREQ packet, it will immediately send a false RREP packet with a modified higher sequence number. So, that the source node assumes that node is having the fresh route towards the destination.

The source node ignores the RREP packet received from other nodes and begins to send the data packets over malicious node. A malicious node takes all the routes towards itself. It does not allow forwarding any packet anywhere. This attack is called a black hole as it swallows all objects and data packets. Cooperative black hole means the malicious nodes act in a group. However, in reality, the packets are consumed by node. For example, source A wants to send packets to destination D. In Figure 6.5, source A initiates the route discovery process. Let M be the malicious node which has no fresh route to destination. Node M claims to have the route to destination and sends reply RREP packet to A. The reply from the malicious node reaches the source node earlier than the reply from the legitimate node, as the malicious node does not have to check its routing cache as the other legitimate nodes. The source chooses the path provided by the malicious node and the data packets are dropped. The malicious node forms a black hole in the network and this problem is called black hole problem.
6.4.7 Avoiding Black Hole Attack

The reputation based mobile agent approach uses a mobile agent to collect the routes from different nodes. Even if there are some malicious nodes in the network, the mobile agents are not affected by them. A mobile agent visits every node in the path to the destination and checks the reputation table in the reverse path. It just adds the reputation value of the nodes in the reverse path in the source list and returns to the source. Here there is no chance for any malicious node to do alterations in the routes of the mobile agent. Also there is no chance for any malicious node for giving a reply to the source. Source will accept only the replies from the mobile agent.
6.4.8 Byzantine attack

A compromised intermediate node works alone, or a set of compromised intermediate nodes works in collusion and carry out attacks such as creating routing loops, forwarding packets through non-optimal paths, or selectively dropping packets, which results in disruption or degradation of the routing services. This is explained by Awerbuch et al., (2002) in more detail with respect to on-demand protocols.

6.4.9 Avoiding Byzantine attack

The reputation based mobile agent approach uses the mobile agent for gathering the routes and uses the reputation mechanism to classify the nodes as well behaving and misbehaving. Nodes that are not maintaining a higher reputation value will not receive the network services. Thus the compromised nodes in single or in groups can not affect the routing procedure.

6.4.10 DoS Attack

DoS attack refers to denial of service attack. Here a valid node is denied from getting the network services. There are multiple ways of doing such attack. For example false advertisement message carrying information about a well behaving node may make other nodes in the network to deny services to that node suspecting it as misbehaving. A simple selfish behaviour of a node can also be considered as a DoS attack, since a valid mobile node cannot use the network services. Other than listed above there is various ways in which a valid node may be denied network services. For example selectively dropping Route Request packets by a selfish node may lead a well behaving node to spend most of its time in route discovery process rather than sending or forwarding data packets. Whenever a valid node is not able to receive the network service we can call it as denial of service attack.
6.4.11 Avoiding DoS attacks

The method used in this thesis enters a node in black list (denying service) only when it understands misbehaviour of that node by direct monitoring, not from the alert/advertisement messages from other nodes. Thus this type of DoS attack is not possible. A good reputation value has to be maintained for receiving network services; hence a malicious/selfish node dropping the packets selectively or fully will not help the mobile nodes for receiving network services. Mobile agent is collecting the routing and reputation information from the nodes; hence there is no possibility to inject malicious nodes in the routes. Thus reputation mechanism and mobile agents are again used for avoiding the DoS attacks.

6.5 SIMULATION RESULTS

NS2 is used for simulation. Performance of normal DSR and the DSR with reputation mechanism using mobile agent is compared. The parameters used for performance analysis are 1. Time taken for route discovery 2. Reliability of the network when number of Route Requests increases 3. Reliability of the route in a network with misbehaving nodes, 4. Packet Delivery Ratio in presence misbehaving of nodes and 5. Success Ratio.

6.5.1 Simulation Setup

The simulated network consists of wireless nodes 20-100 deployed in a field of 1200 x 1200 square meters. The random waypoint is chosen as a mobility model. Each node is first randomly placed in the field, waits for the pause time (10 second in our simulation), then moves to another random position with a speed chosen between 1 to 15 m/s. The constant bit rate (CBR) traffic is selected as the traffic model. Each simulation is run for
900 seconds. Table 6.1 shows the static parameters set in the simulation. Table 6.2 shows some of the parameters that are varied during simulation.

Table 6.1 Static Network parameters for simulation of the approach

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mobility model</td>
<td>Random Way point</td>
</tr>
<tr>
<td>2</td>
<td>MAC</td>
<td>IEEE 802.11</td>
</tr>
<tr>
<td>3</td>
<td>Antenna</td>
<td>Omni Directional</td>
</tr>
<tr>
<td>4</td>
<td>Network Grid</td>
<td>1200 X 1200</td>
</tr>
<tr>
<td>5</td>
<td>Total Simulation Time</td>
<td>900 s</td>
</tr>
<tr>
<td>6</td>
<td>Traffic</td>
<td>CBR</td>
</tr>
<tr>
<td>7</td>
<td>Packet Size</td>
<td>64 bytes</td>
</tr>
</tbody>
</table>

Table 6.2 Variable Network parameters for simulation of the approach

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Minimum</td>
</tr>
<tr>
<td>1</td>
<td>Number of Nodes</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>Node Speed</td>
<td>10 m/s</td>
</tr>
<tr>
<td>3</td>
<td>Pause time</td>
<td>100 ms</td>
</tr>
<tr>
<td>4</td>
<td>Number of Source destination pairs</td>
<td>2</td>
</tr>
</tbody>
</table>
6.5.2 The Network Model

Following are the assumptions and network model used in this approach.

1. Each node is identified by a unique, persistent ID.
2. Each node runs a “Watchdog” mechanism to monitor other nodes.
3. Network is dense enough to establish communications.
4. Links are bidirectional, i.e. If communication between A to B is possible, then communication between B to A is also possible.
5. Some nodes are misbehaving, selfish or malicious.
7. There is no collision in the signals.
8. Mobile agents can be authenticated by using any of the existing authentication mechanism.
9. Reputation table maintained at each node is tamper proof.

Figure 6.6 shows the time taken for route discovery when the number of Route Requests is varied. The number of nodes in the scenario is set to 75. This scenario assumes no misbehaving nodes. Mobility of the nodes is set to 12 m/s. The time taken is more for DSR when number of requests is increased, but for Reputation based DSR with mobile agents, it is comparatively less. In this approach, the mobile agent takes care of gathering the route, so the time taken to receive a reply by the node is less. The mobile agents will move independently in the network and collect routes. They go to each node, check whether already visited, otherwise add that node in the list and move on. In the return path, they add reputation values of the neighbouring nodes as seen by this node and finally reach the source with a stack of nodes to reach the destination.
Figure 6.6 Time taken for Route Discovery Vs. Number of Route Requests

Figure 6.7 Time taken for Route Discovery Vs. Number of Malicious nodes
Figure 6.7 shows the time taken for receiving a successful Route Reply at the source after sending the Route Request packet in presence of malicious nodes. The malicious nodes will drop the Route Request packets or misdirect them randomly. The time taken in reputation based DSR using mobile agent is less compared to normal DSR. The presence or absence of malicious nodes does not make any change in the reputation based DSR with mobile agents, whereas the normal DSR is worse affected by malicious nodes. The time taken for route discovery is between 10-50 ms for reputation based approach using mobile agent. For normal DSR it is between 30 and 300ms. In normal DSR the presence of malicious nodes drastically affects the performance.

Figure 6.8 gives the throughput of the network when the number of Route Requests increases. The number of nodes is set to 75. Number of malicious nodes is 10. Malicious nodes are configured to randomly drop some Route Request and data packets. As the traffic increases, the throughput decreases in DSR protocol. But in reputation based DSR, it manages to be same. There may be misbehaving nodes in the network, which drops the routing or data packets of other nodes. This reduces throughput of the network. But reputation based approach using mobile agent identifies a trustworthy route with less number of misbehaving nodes. Thus the throughput of this approach is higher compared to normal DSR.
Figure 6.8 Throughput Vs Number of Route Requests

Figure 6.9 Throughput of the network when number of misbehaving nodes are increased.
Figure 6.9 shows, that even when the number of malicious nodes is increased, the R-DSR is able to provide reliable communication. This is because R-DSR selects the best route from the available route replies based on the reputation value. Thus from the results it is proved that R-DSR provides better performance compared to DSR. When a node receives a data packet to forward to its neighbour, it checks the reputation table for the reputation value of the sender of the packet. If the reputation value is greater than the threshold value 0.8, it forwards the packet, else it discards the packet. By this method the nodes are forced to forward others’ packet, else their reputation value will drop, leading to discarding of their packets. Thus a node which is selfish is denied from network services.

Figure 6.10 shows the Packet Delivery Ratio (PDR) of the network when number of misbehaving nodes are increased. PDR is calculated as the ratio of total number of packets received to total number of packets sent at application layer. The PDR is more for reputation based DSR protocol using mobile agent compared to standard DSR. Reputation based DSR uses the trustworthy route by avoiding misbehaving nodes, hence number of packet drops are less. But DSR selects the route which has misbehaving nodes, hence packet drops are more.

Figure 6.11 shows the packet delivery ratio by varying the pause time. It shows that reputation based DSR outperforms basic DSR.
Figure 6.10 Packet Delivery Ratio of the network when number of misbehaving nodes are increased.

Figure 6.11 Packet Delivery Ratio Vs Pause time in seconds
Figure 6.12 Success ratio Vs number of nodes

Figure 6.12 shows the success ratio for DSR and reputation based DSR. Success ratio is the ratio of number of Route Requests sent to number of route replies received. Success ratio is more for reputation based DSR and less for DSR. The presence of malicious nodes will decrease the success ratio in normal DSR. Malicious nodes will drop the Route Request packets or they may redirect the Route Request packets. Normal DSR can not handle this problem. But reputation based DSR with mobile agents handles this problem. Mobile agents will help to avoid the attacks by malicious nodes.

Figure 6.13 shows the success ratio for various number of Route Requests. Here also reputation based DSR outperforms DSR.
Figure 6.13 Success ratio Vs Number of Route Requests

Figure 6.14 Success ratio Vs Number of Malicious nodes
Figure 6.14 gives the success ratio when the number of malicious nodes is increased. The total number of nodes is set as 75. Reputation based DSR with mobile agent approach outperforms normal DSR. The success ratio determines how many successful route replies are received by a node. Only if a successful Route Reply is received, a node can start its communication. Thus success ratio is an important factor which determines the performance of any routing protocol.

This approach has several advantages.

1. The path used for data communication is trustworthy.
2. Communication overhead is less.
3. Increases throughput in the network.
4. Implementation is simple since it needs only a little change in DSR algorithm.
5. Avoids misbehaving nodes from receiving network services.
6. Avoids some of the routing attacks specific to on-demand protocols.

6.6 ANALYSIS OF THE APPROACH

This section gives some case studies.

Case 1: A malicious node may discard the Route Requests or redirect the Route Requests or give a wrong Route Reply including itself in the source list, or a Route Reply which does not even exist.

All these problems can not occur, because our approach uses a mobile agent for finding the routes.
Case 2: A node can participate in route discovery process but deny forwarding the data packets of another node.

A mobile agent is used in route discovery process; hence a trustworthy route is available.

Case 3: A node accepts its own data and drops others’ data.

Reputation mechanism identifies misbehaving nodes and isolates them from the network.

Case 4: A malicious node enters a higher or lower reputation value for its neighbour for doing a collaborative attack.

It is assumed that the reputation table is tamper proof and can be updated automatically only by the software.

### 6.7 STEPS AT EACH NODE

This section gives the steps done at each node.

**Source node:**

Step 1: If a node has data to transmit, check route cache.

Step 2: If a fresh route is available, then use it to transmit data packets to the destination.

Step 3: Else, start the mobile agent and pass to all neighbours. Mobile agent carries a source id, destination id, mobile agent id, and a stack.

Step 4: Wait for replies from more than one mobile agent.
Step 5: If the mobile agent carries no reputation values corresponding to the nodes, drop the reply.

Step 6: Else calculate the average value for each route that is gathered by the mobile agents.

Step 7: Select a route with highest reputation value

Step 8: Transfer data through this route.

**Mobile Agent**: (Mobile agent starts traversing to the next node from the source node based on the neighbour list of the source node. Number of copies of mobile agent is equal to the number of neighbours of source node. Each mobile agent has a unique identifier and the parameters as listed in step 3 of source node)

Step 1: Move autonomously from source node to one of the next hop neighbour.

Step 2: If the visited node is a new node add it to the stack and be locally broadcasted to all the neighbours of this node.

Step 3: Else if it is a node that is already visited then silently die.

Step 4: Else if this node is the destination, then check the reputation value of the last visited node from the reputation table of the destination and make an entry in the stack against the field available in it.

Step 5: Repeat steps 6 and 7 until source node is reached.

Step 6: Move to the previous hop as specified in the stack.
Step 7: On reaching the previous hop note the reputation value of the next hop as seen by this node.

Step 8: If the visited node is the source node and if the reputation values are filled in, enter the collected route (stack) in the route cache of the source and stop.

The destination nodes and the intermediate nodes need not do anything. They just allow the mobile agent to execute its code in them.

6.8 OPTIMIZATION OVER THE APPROACH

The approach simulated here can be optimized in the following manner.

6.8.1 Hop Count

The mobile agent starts when the source node initiates the route discovery process. It dies when it reaches the source using the reverse path from the destination. What happens if the mobile agent is not able to find out the destination? Mobile agent will be just moving on in the network. In order to avoid problems like this, we can add a hop count as a parameter in the mobile agent. The mobile agent will reduce the hop count by one whenever it reaches a new node, and die when it becomes zero. This avoids the problem of searching for unreachable destinations. In our simulation we have modified the DSR Route Request message to act as a mobile agent, thus when Time to Live expires the mobile agent will expire.
6.8.2 Using Reputation Values

In the simulations, mobile agent moves to all the neighbours of the source node. In order to get a trustworthy route, the mobile agent can move only to the nodes with high reputation value (instead on moving to all the neighbours.) That is it can move to all the nodes in the white list alone. This will increase the chances of getting a trustworthy route and reduce the overhead. Similarly while being locally broadcasted from intermediate node; it can check the reputation values and copies of mobile agent can be sent only to the neighbours with high reputation value. In this simulation, we have locally broadcasted to all the nodes just to increase the chances of getting more replies.

6.8.3 Selecting next node

The mobile agent on reaching the next hop, after adding its id to the stack, instead of being broadcasted to all nodes, or instead of moving only to neighbours with higher reputation value, can select one hop from the neighbours list and move to that hop. Another mobile agent which enters the same intermediate node can select another node from the neighbour list or white list and move on. But some mechanism should be there to uniquely select one neighbour from the neighbour list. That is one mobile agent should not select the same neighbour as its next hop. To achieve this either the mobile agents can communicate with each other and inform to each other about which node is selected as a next hop, or the intermediate nodes can have a field in their neighbours table which can be set by the mobile agents on selecting a particular neighbour.
6.9 COMPARISON WITH OTHER APPROACHES

Table 6.3 Comparison of various approaches in terms of attack handling

<table>
<thead>
<tr>
<th>Approaches</th>
<th>ARAN</th>
<th>ARIADNE</th>
<th>CONFIDANT</th>
<th>CORE</th>
<th>THIS APPROACH</th>
</tr>
</thead>
<tbody>
<tr>
<td>DoS</td>
<td>X</td>
<td>√</td>
<td>X</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td>Blackhole</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rushing</td>
<td></td>
<td>^</td>
<td>X</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td>Wormhole</td>
<td>X</td>
<td>^</td>
<td>X</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td>Selfish</td>
<td>X</td>
<td>X</td>
<td>√</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\( \checkmark \) Attacks handled  \( X \) Attacks not handled  \( ^\) Can handle with some improvements

Table 6.3 gives a comparison on the approach used in this thesis with some of the existing approaches in terms of attack handling. These approaches were dealt in detail in chapter 2 of this thesis. ARAN uses digital signatures to provide authentication to mobile nodes and it is capable of handling rushing attacks alone. It cannot provide solutions for black hole and other related routing attacks.

ARIADNE is an approach based on cryptographic primitives. It mainly concentrates on routing misbehaviour and application layer attacks. It does not deal with selfish nodes. It can handle rushing and wormhole attacks by making some improvements. CORE and CONFIDANT can deal with selfish and some kinds of malicious attacks. Both cannot handle rushing, wormhole and DoS attacks. Table 6.4 gives a comparison of the approach in this thesis with some of the important existing routing protocols.
Table 6.4 Comparison with other routing protocols

<table>
<thead>
<tr>
<th>Methods</th>
<th>Main Idea</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
</table>
| ARAN | Uses digital certificates to provide authentication, non repudiation and integrity | Handles routing attacks | Does not deal with selfish nodes  
Computational cost is high due to the exchange of digital certificates.  
Need for a centralized server for issuing certificates. |
| Reputed ARAN | Uses digital certificates for authentication and reputation mechanism to detect selfish nodes. Reputation is modified based on whether a node successfully forwards a packet or not. | Controls selfishness and also handles routing attacks | Use of digital certificates increases communication overhead  
Use of special acknowledgements for every data packet received at the destination increases overhead  
Needs a trusted certificate server. |
| ARIADNE | Uses symmetric cryptographic primitives and it is based on TESLA | Handles application layer attacks and some of the routing attacks | Needs a Key Distribution Center  
Communication overhead is more |
| CONFIDANT | Reputation based mechanism | Path manager selects a path by avoiding malicious nodes in the network | Does not account routing misbehaviour.  
Does not deal how to calculate reputation values properly  
False alarm messages may cause problems to well behaving nodes. |
| CORE | Reputation value is used to make decisions about cooperation or gradual isolation of a node | Identifies and isolates selfish nodes | Nodes may build up credits and start misbehaving  
Does not account routing misbehaviours |
| Mobile Agent Based Detection of Selfish Node in MANET | Uses mobile agents to detect selfish nodes. When the source detects drop of packets, it send set of mobile agents and the mobile agent calculate packet delivery ratio, using this it detects selfish nodes. | Detects selfish nodes | Does not consider routing attacks. |
| This Approach | Uses mobile agents and reputation mechanism to detect and isolate misbehaving nodes | Detects and isolates selfish as well as malicious nodes.  
Avoids routing attacks like wormhole, blackhole, grayhole, byzantine etc. | Does not include any mechanism for authentication and integrity. |
6.10 SUMMARY

Thus this chapter has explained the main contribution of the thesis. The reputation based routing protocol using mobile agent. The new routing protocol is implemented over the existing DSR protocol, which is an on-demand routing protocol. This chapter explained the importance of mobile agent which is used to avoid many routing attacks like black hole attack, wormhole attack, gray-hole attack etc. Many of the existing approaches in the literature are not successful in avoiding or mitigating the effects of both packet forwarding and route disturbing misbehaviours. But the approach proposed in this thesis succeeds in reducing the data forwarding misbehaviour by using the reputation mechanism and reduces the routing attacks by using the mobile agents. Thus our approach becomes a complete secure routing protocol for mobile ad-hoc networks.