4. WORK PLAN & METHODOLOGY

4.1. CORE IMPLEMENTATION

4.1.1. Basic Protocol Simulation

This chapter discusses how the AODV protocol is simulated and implemented. First the platform i.e. Fedora 8 was set up in a virtual environment. Then NS2 is set up on the platform on which the above said protocols will be implemented. NS2 requires a script file to be run on it. These script files are written in a language called TCL (Tool Command Language). We have make use of shell scripting &Gnuplot for plotting of graphs.

4.1.2. QoS-Enabled Protocol Simulation

In this proposed research work a quality of service (QoS) architecture for supporting real-time data transmission in mobile ad hoc networks (MANETs) is explored. The QoS architecture includes a QoS transport layer, QoS routing, queue management and a priority MAC protocol. Through simulations, it is found that the QoS architecture reduces packet loss and greatly improves the resource utilization in MANETs.

Providing support for real-time data transmission is an important yet challenging goal for MANETs. Much research has been done in each network layer to support real-time data transmission. Various routing protocols have been proposed for routing a data from source to destination. But these protocols do not provide admission control or find a path with large enough bandwidth to support a given request. Data communication is the result of each network layer’s effort; thus, the cooperation of all network layers is needed to provide QoS support. However, these designs are not comprehensive enough to include all the networking layers. Therefore, the QoS architecture proposed in this section supports real-time data transmission. It extends from the application layer to the MAC layer to. This QoS architecture is described in the following section.

4.2. QOS ARCHITECTURE

In Figure 4.1, we show our proposed Qo$S$ architecture, which includes all networking layers from the application layer to the MAC layer. The bold lines indicate the flow of data packets and the narrow lines indicate the flow of control packets. Each layer’s features are detailed below.
4.2.1. Application Layer:

Applications can be categorized into real-time and non-real-time applications based on their sensitivity to packet delay. Real-time applications have strict requirements on the packet delay. Therefore, packet retransmission is not allowed. The applications that fit into this category are on-line live movies and video conferencing. Many video compression technologies, such as MPEG-4, H.263, and multiple-description coding, can compress video with different coding rates to meet different channel conditions. In addition, most of these compression schemes have error resilience features to recover the video frame, if some packets are lost. Thus, choosing the right coding rate to compress the video is important, and some reasonable packet loss is acceptable. For this purpose, we need an efficient model which highly improves the packet delivery ratio. In this thesis, an attempt is made to achieve high efficiency using “Bandwidth Estimation” technique. On the other hand, for non-real-time applications such as Email and FTP, packet delay is not a big issue, and packet delivery is guaranteed by explicit acknowledgements in the transport layer.
4.2.2. Transport Layer:

UDP and TCP are two transport layer protocols widely used in wired networks. UDP has no congestion control scheme to react to network congestion. Applications that use UDP as the underlying transport protocol to transmit packets can easily overwhelm the network with data, which results in a considerable amount of wasted energy and bandwidth in transmitting packets that will be dropped due to congestion. Therefore, some pre-dropping of UDP packets should be investigated to react to congestion. TCP has an inherent congestion control scheme, so congestion control is not a problem. However, TCP’s performance should be optimized to adjust the TCP window, which requires feedback information from the lower network layers. Therefore, some information from the packet queue and the routing layer should be sent to the transport layer for performance optimization.

4.2.3. Network layer

To support QoS, the routing protocol should have an embedded scheme such as call admission or adaptive feedback that is designed to support QoS. At the same time, non-QoS-aware routing that is targeted at finding a feasible path should be offered as well. For QoS-aware routing, information about the current network status is provided to the application for performance optimization. Also, the routing layer should get enough channel information from the lower layers so that the admission/adaptive scheme can be performed based on the network status. Therefore, two cross-layer designs should be implemented in QoS-aware routing. One is to obtain the network resource information from lower layers, and the other is to send the network status to the applications. To offer QoS to the applications, resource reservation should be incorporated. An RSVP-type signaling scheme is not desirable in MANETs due to its high overhead. Therefore, in-band and soft resource reservation (i.e., best effort rather than guaranteed reservations) should be done during the route discovery phase and during route maintenance. The transmissions that occur between the break down of old routes and the set up of new routes will severely affect the QoS provided by the network. Therefore, some prediction of route breaks should be incorporated.

Overall, QoS-aware routing should have the following features that traditional routing does not support:

a) Obtain resource information from lower layers;

b) Offer bandwidth information to applications;

c) Incorporate resource reservation schemes;
4.2.4. Link layer

The link layer needs to discriminate the different priority packets and schedule packet delivery according to priority levels. The service differentiation should be completed in the packet queue through queue management and in the MAC layer through a MAC discriminator and priority classifier.

a) Queue Management: The aim of queue management is to schedule the different priority packets. Real-time data should have higher priority to be sent to the channel compared with packets such as FTP and Email. Therefore, real-time data will be put in front of the non-real-time data in the packet queue. When the network is congested, the last packet in the packet queue will be dropped. Therefore, incorporating queue management will reduce the possibility that real-time packets are dropped in the packet queue when the network is congested. Thus, packet delivery ratio can be improved. Also, the packets whose delay has already exceeded the applications requirement should be eliminated from the packet queue before transmission to save the transmission of packets that will be useless to the receiver. If different flows go through the same host, it is easier to do the priority regulation in the packet queue than in the MAC layer.

b) MAC Discriminator: The main function of the MAC discriminator is to differentiate data packets and control packets that arrive from the wireless channel. Data packets are sent to the network layer; ARP (address resolution protocol) packets go to the queue directly; MAC packets, such as the RTS, CTS, and ACK packets used in IEEE 802.11, stay in the MAC layer; and the bandwidth estimation control packets are sent to the bandwidth estimation module for use in the routing layer’s admission/adaptive scheme.

c) Priority Classifier and Packet Scheduler: To offer service differentiation in a distributed ad hoc network, real-time packets should be granted higher priority to capture the channel. The priority classifier differentiates the different data packets that arrive from the packet queue and directs the packet scheduler to schedule the packet delivery based on the priority level of the current packet.

4.3. BANDWIDTH ESTIMATION

In a distributed ad hoc network, a host’s available bandwidth is not only decided by the raw channel bandwidth, but also by its neighbor’s bandwidth usage and interference caused by other sources, each of which reduces a host’s available bandwidth for
Performace Evaluation of Ad Hoc Networking Protocol with QoS (Quality of Service)

transmitting data. Therefore, applications cannot properly optimize their coding rate without knowledge of the status of the entire network. Thus, bandwidth estimation is a fundamental function that is needed to provide QoS in MANETs. However, bandwidth estimation is extremely difficult, because each host has imprecise knowledge of the network status and links change dynamically. Therefore, an effective bandwidth estimation scheme is highly desirable. Bandwidth estimation can be done using various methods; the one of the way of bandwidth estimation is a cross-layer design of the routing and MAC layers, and in second the available bandwidth is estimated in the MAC layer and is sent to the routing layer for admission control. Therefore, bandwidth estimation can be performed in several different network layers, as shown in Fig 4.1.

Main aim is to improve QoS with major focus on Bandwidth parameter. Fig. below shows the RREQ message format used in AODV protocol. For enhancing performance of the basic protocol one more field named “Bandwidth Required” is added in the given RREQ format. The RREQ message contains following information: message type, source address, destination address, broadcast ID, hop count, source sequence number, destination sequence number, request time (timestamp).

Whenever the source node issues a new RREQ, the broadcast ID is incremented by one. Thus, the source and destination addresses, together with the broadcast ID, uniquely identify this RREQ packet. The source node broadcasts the RREQ to all nodes within its transmission range. These neighboring nodes will then pass on the RREQ to other nodes in the same manner. As the RREQ is broadcasted in the whole network, some nodes may receive several copies of the same RREQ. When an intermediate node receives a RREQ, the node checks if already received a RREQ with the same broadcast id and source address. The node cashes broadcast id and source address for first time and drops redundant RREQ messages. When the destination node receives first route request message, it generates so called reverse request (R-RREQ) message and broadcasts it to neighbor nodes within transmission range like the RREQ of source node does.

This RREQ packet is used to store the information of bandwidth required field & then used to compare it with the current requirement. And, the packet is forwarded to the next intermediate node only when it does have sufficient amount of bandwidth otherwise it is dropped & then it is re-transmitted.
Performance Evaluation of Ad Hoc Networking Protocol with QoS (Quality of Service)

Fig 4.2 & 4.3 show RREQ Message Format before & after QoS-Enabling.

<table>
<thead>
<tr>
<th>TYPE</th>
<th>RESERVED</th>
<th>HOP COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broadcast ID</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination IP Address</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination Sequence Number</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source IP Address</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source Sequence Number</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Request Time</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig 4.2 : RREQ Message Format before QoS-Enabling

<table>
<thead>
<tr>
<th>TYPE</th>
<th>RESERVED</th>
<th>HOP COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bandwidth Required</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Broadcast ID</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination IP Address</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination Sequence Number</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source IP Address</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source Sequence Number</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Request Time</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig 4.3 : RREQ Message Format after QoS-Enabling

4.4. TEST PROCEDURE (BUILDING & TESTING)

4.4.1. Performance Analysis

The performance analysis can be done on Fedora 8 as the Operating System. NS 2.34 was installed on the platform for simulating the protocols along with necessary software such as GnuPlot, which is software for plotting graphs from the trace files. NS (version 2) is an object oriented, discrete event driven network simulator written in C++ and OTcl. It implements network protocols such as TCP and UDP, traffic source behavior such as FTP, Telnet, Web, CBR and VBR, router queue management mechanism such as Drop-Tail, RED and CBQ, routing algorithms such as Dijkstra, AODV, DSR and TORA more. NS also implements multicasting and some of the MAC layer protocols for LAN simulations.
4.4.2. Traffic Environment

The tests will be performed on CBR traffic with 50 nodes. Packet size is set to 500 and the time interval between transferring the packets was set to 0.005 ms. Bit rate is set to 1 Mbps with a Drop Tail of 10 ms. As it is not easy to create traffic simulations for such large number of nodes manually, therefore the simulations are generated with the help of CMU traffic generator and the scenario was generated with the help of setdest, which are the tools preinstalled with the NS2. The field configuration is set to 500 by 500 m.

4.4.3. Performance Metrics used for Analysis

The following metrics will be used for the comparison of the protocols:

1) Throughput: This is the effective share of bandwidth that the application is getting from the network.

2) Bandwidth: This signifies the portion of the available capacity of an end-to-end network path that is accessible to the application or data flow. Consequently, the number of bits that are injected into the network by the various flows of an application have to be adjusted accordingly.

3) Average Packet Delay: Average packet delivery time from a source to a destination. First for each source-destination pair, an average delay for packet delivery is computed. Then the whole average delay is computed from each pair average delay. End-to-end delay includes the delay in the send buffer, the delay in the interface queue, the bandwidth contention delay at the MAC, and the propagation delay.

4) Packet Delivery Ratio: It is a ratio of number of data packets delivered to the destination and the number of data packets sent by the source. Number of Data Packets Delivered over Number of Data Packets Generated. Number of Data Packets Delivered is the total number of received data packets by destinations; Number of Data Packets Generated is the total number of generated data packets by sources. This metric can measure the delivery reliability, the throughput of the protocol.

5) Network Overhead Load: It is the ratio of total amount of overhead caused due to control routing packets and the amount of wireless bandwidth wasted to transmit the packets that are dropped in other links. So we can estimate how many transmitted routing messages are used for one successful data packet delivery by this metric to determine the efficiency and scalability of the protocol.