A NOVEL RESOURCE EFFICIENT DMMS APPROACH FOR NETWORK MONITORING AND CONTROLLING FUNCTIONS

Golam R. Khan[1], Sharmistha Khan[2], Dhadesugoor R. Vaman[3], and Suxia Cui[4]

Department of Electrical and Computer Engineering, Prairie View A&M University, Prairie View, Texas 77446, USA

ABSTRACT

In this paper, we propose a novel Distributed MANET Management System (DMMS) approach to use cross layer models to demonstrate a simplified way of efficiently managing the overall performance of individual network resources (nodes) and the network itself which is critical for not only monitoring the traffic, but also dynamically controlling the end-to-end Quality of Service (QoS) for different applications. In the proposed DMMS architecture, each network resource maintains a set of Management Information Base (MIB) elements and stores resource activities in their abstraction in terms of counters, timer, flag and threshold values. The abstract data is exchanged between different management agents residing in different resources on a need-to-know basis and each agent logically executes management functions locally to develop understanding of the behavior of all network resources to ensure that user protocols can function smoothly. However, in traditional network management systems, they collect statistical data such as resource usage and performance by spoofing of resources. The amount of data that is exchanged with other resources through management protocols that can be extremely high and the bandwidth for overhead management functions increases significantly. Also, the data storage requirements in each network resource for management functions increases and become inefficient as it increases the power usage for processing. Our proposed scheme targets at solving the problems.

KEYWORDS

Network Management, MANET, Distributed Network Management System, MIB, Managed Objects (MO)

1.INTRODUCTION

Network management is defined as “a process of controlling a complex data network so as to maximize its efficiency and productivity” [1]. This process involves data collection, data processing, data analysis, fault identification and fault clearing methods in order to support desired multi-service provisioning end-to-end with performance assurance needed. To accomplish this process, the International Standards Organization (ISO) referred to as the OSI management model [1, 2], defines five areas of network management. They include fault management, accounting management, configuration management, performance management, and security management. In Fig. 1, the flow diagram of the management functions is given. At any instance of time, the operational controls of the network resources are in the hands of these management functions. When the network is operating in normal mode, the control is in the hands of performance management. If any resource experiences a fault, the fault management controls the resource. The configuration management takes control whenever a new resource is being configured, the configuration management takes control, the security management takes control if any resource has a security violation, and the accounting management controls the resource cost and usage management. These functional management components are designed modularly and each of these functions looks at the MIB elements to take any actions whether it is real time or non-real time. Thus, some abstraction is maintained to simplify the management functionality of the network.

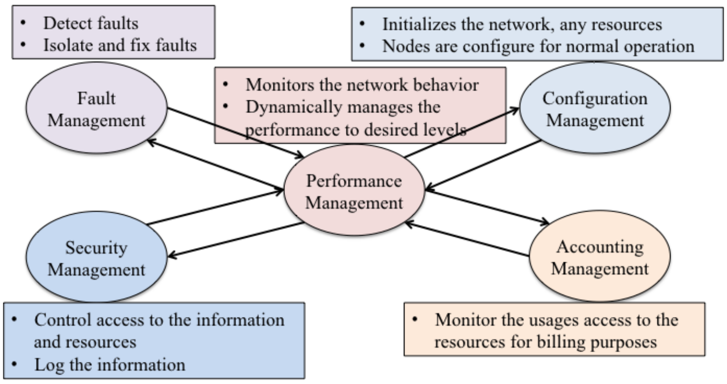

Figure 1. Flow diagram of the management functions

The design of network management architectures uses either centralized where a single network manager station or hierarchical where it maintain multi-layered network manager or distributed that allow each node to have an agent which takes manager-ship when initiating an action or responds to a command. Thus, it is important to use distributed network management where a management agent in a node takes manager-ship for a small duration of time and relinquished once the execution of an activity is done. This allows every node to be acting as manager at different instances of time.

This leads to the design of distributed network management where the users can measure the behavior of the network for their own applications without requiring the network to exchange such data is very crucial as more and more heterogeneous wireless networks are being connected. The end user devices must be able to develop management entities that allow computation of traffic and provide real time control will allow performance assurance of multi-service provisioning. This paper address the distributed management architecture and demonstrate the cost effective computation of the management entities at each user for both performance monitoring and real time control of bounded performance management of applications supported for each source-destination pair of nodes in Mobile Ad Hoc Networks (MANET).

In our previous work, we have shown the overview of a resource efficient Distributed MANET Management System (DMMS) architecture [10]. In this paper, we are going to demonstrate the cost effective computation of the management entities at each user for monitoring the overall network such as user data throughput and delay, and real time network control functions where we have shown different management aspects for different multi-service applications such as Position, Location and Tracking (PL&T) system [11] and Graph Theoretical Routing application [12] with minimal use of bandwidth in the network for management functions. The remaining sections are organized as follows:

- Section 2 Background Research
- Section 3 Overview of Distributed MANET Management System (DMMS) architecture
- Section 4 Management Functions
- Section 5 shows a real time simulation for DMMS where Network Simulator-2 (NS-2) and MATLAB are used as simulation tools. Through the simulation results, it will analyze the performance of the proposed DMMS.
- Section 6 Conclusions

2. BACKGROUND RESEARCH

In Table 1 we have shown the summary of some related works.

Table 1.Summary of different network management system architectures

Network Management System	Main Management Functionalities	Limitations
Ad hoc Network Management Protocol (ANMP) [3]	• Hierarchical NMA is used. • Reduce the number of messages exchanged between the manager and the agents. • Focused on data collection, configuration/fault management, and security management • SNMP is used.	• Performance management function is not developed. • Cost of maintaining a hierarchy in the face of node mobility • Cluster maintenance algorithms runs periodically
Resource Monitoring Issues in Ad Hoc Networks [4]	• A small-scale (13 nodes) ad hoc network is implemented and overall performance is evaluated • Run in a highly mobile ad hoc network	• Periodically broadcasting the beacons that causes high message overhead • Not saleable
Network Management System for Large-scale WSNs [5]	• Centralized NMA • Overcome the bad impact of polling and periodically broadcasting • Configuration management, performance management, fault management and accounting management is considered	• One manager • High communicate overhead for data polling and complex to implement • Focuses on the performance analyses of communication from the node to the gateway
Network Management Framework for WSNs [6]	• Cluster-based hierarchical architecture is used • Defined the MIB according to the roles of nodes and characteristics of information • Reduces the management overhead of the network	• Not shown the computation of the aggregated or statistical information of the network • Not saleable. • Complex to implement and not flexible enough
The Guerrilla Management Architecture for Ad hoc Networks [7]	• Adaptive and autonomous management of ad hoc networks • Nodes range in heterogeneous functionality and capability • Scalable	• Cluster heads poll management information from the nodes in a centralized manner that introduce an additional overhead that will increase energy consumption and decrease the available bandwidth
Towards Autonomic Network Performance Management in Mobile Ad Hoc Networks [8]	• Focused on centralized implementation • Designed to monitor, model, optimize, and configure the controllable node • Tuning the performance of the overall network performance.	• Not mansions about the fully distributed or hybrid approach
MANNA: Management Architecture for Wireless Sensor Networks [9]	• Provides self management and self configuration • Policy-based management framework, • Coverage maintenance, and fault detection	• Polling management messages sometimes causes huge amount of management traffic for a large network.

From the literature reviews that are discussed above, it is clear that, all the proposed network management architectures have some features that are really beneficial to manage a network. However, still there are some major drawbacks that also need to be considered. Most of the proposed systems [3-9] want to reduce management message overhead, provide power efficiency and scalability for managing large-scale networks where SNMP is used for exchanging MIBs. Thus, that has not given the flexibility to customize packet structure as SNMP has a fixed packet structure. Again, most of the cases, management packets are periodically broadcasted that reduces resource efficiency and bandwidth efficiency as well. Furthermore, for collecting statistical data, spoofing of resources are used where it requires exchanging data with other resources through management protocols that ultimately increases the bandwidth for overhead management functions. Considering these issues, we have developed such a distributed network management system that ensure flexibility of using network resources, bandwidth efficiency and as well as resource efficiency. In our proposed DMMS, we have shown a simplified way to find the network performance by using cross layer management objects (MOs) and distributed the management functions among all the cluster members.

3. OVERVIEW OF DMMS ARCHITECTURE

We have shows the architectural representation of Distributed MANET Management System (DMMS) Architecture that was shown in [10]. In Fig. 2 shows the DMMS architecture where each node has an agent with an associated distributed MANET management system MIB (dMmsMIB). Any agent can assume managership at an instance of time. The network management operation is based upon one of the agents to assume managership (become manager) and communicate with a remote agent for accessing the dMmsMIB elements for either monitoring of the network behavior or for controlling the network behavior in real time.

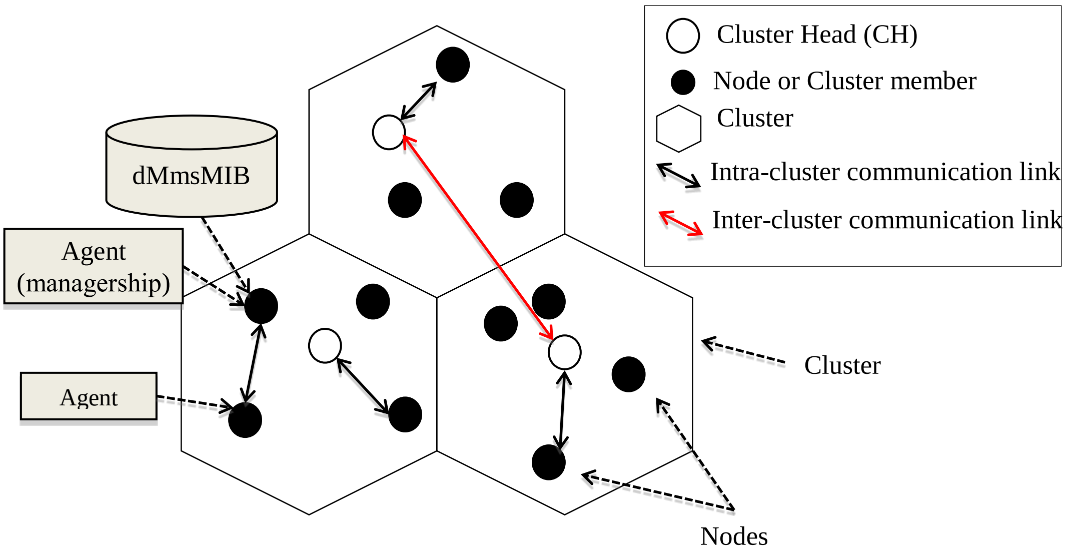

Figure 2. Architecture of the Distributed MANET Management System (DMMS)

Figure 3 shows a block schematic diagram of two nodes with network management function with associated MIB and OSI User Protocol Stack. The User Plane where User Protocol stack resides is used for message exchange between end user applications. The network management function and the associated MIB reside in the management plane. The network management function exchanges the LME data from each user protocol layer that is updated in the associated Attributes. Basically, the Attribute content specifies the behavior of the user functions at the local protocol layers. This is done at each node. Once this is established, any network management function at any node can access the content of the MIB elements of the network management function at any remote node and compute the necessary controls for execution. Since only MIB elements are exchanged in the management plane, the bandwidth required for control functions can be significantly reduced in the management plane as opposed to sending the strings of data in the user plane.

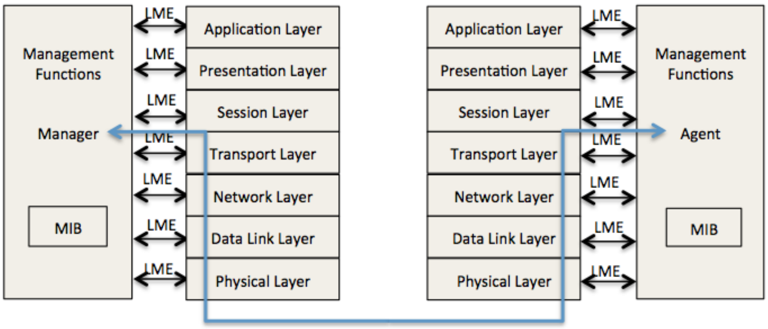

Figure 3. Management Information Exchange

At any instance of time, the network management function at any node can act as a manager and a network management function at a remote node can act as an agent. The manager-agent relationship can be reversed at different instances of time. An agent acting as a manager will issue commands to access the MIB elements of a remote node and the agent at the remote node will respond to the commands. In the simulation of DMMS, the network management function is set up such that each command shall have a response from the remote agent. The communication between two network management functions allows exchanging MIB elements or allows sending commands to operate on the remote MOs and their associated attributes. It is independent of the user protocol. The MIB elements in each node are organized such that a Managed Object (MO) is specified for each user protocol layer with a set of associated attributes. The attributes are set up as counters with content in terms of integer values, flags and status with content as Boolean.
Figure 3 shows the management information exchange process. The simulation is set up such that the management packets are interleaving with user packets within each ensemble. The number of management packets within each ensemble is set based on the quantity of management attributes and the remaining packets assigned as user packets.

The management commands are identified in the management packet as a PDU types. In the proposed design, we have used the following PDU types for Management Information Exchange: *SET*, *SETBULK*, *ACK*, *GET*, *GETBULK*, and *RESPONSE*. These are used for either getting the contents of the elements or operating on the remote elements (MOs and associated Attributes).

The management commands and corresponding actions are specified as follows:

- *GET* and *GETBULK* are used when a management function at a node becomes a manager and requests access to the contents of MOs and associated Attributes at a remote MIB controlled by remote agent at a remote node.
- *RESPONSE* is sent by the remote agent as a response to *GET* and *GETBULK* request.
- *SET* and *SETBULK* are used when a management function at a node wishes to operate on the contents of MOs and associated Attributes at a remote MIB.
- ACK is used to respond to *SET* and *SETBULK* requests identifying the actions taken on the MOs and the attributes at the remote agent.

Figure 4 shows some example of MIBs that are proposed to use as the distributed MANET management system's MIB (dMmsMIB).

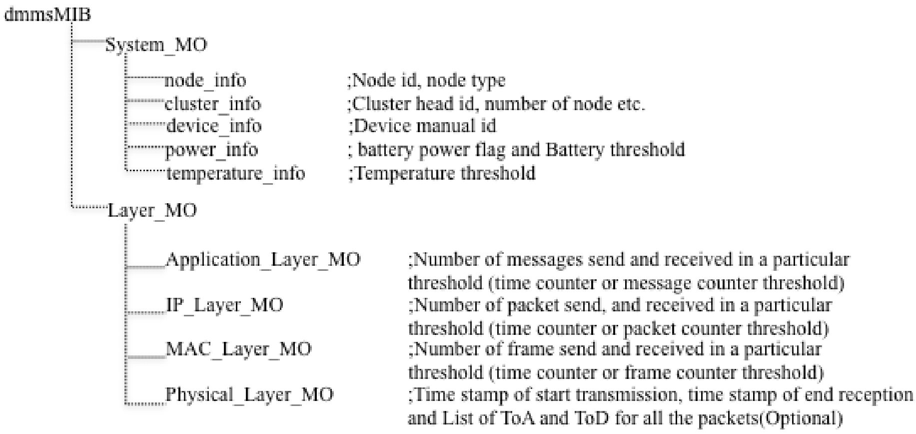

Figure 4. Distributed MANET management system's MIBs (dMmsMIBs)

In [10], we defined two types of management operations such as normal basis management and need-to-know basis management. In normal basis management operation, dynamic MIBs are exchanged. In need-to-know basis management operation, both dynamic and static information are exchanged.

During any transmission, when a node observes that an attribute (e.g. number of packet count) has reached to a threshold, it will send a broadcast message or a unicast message (management) like "*SET_BULK* PDU" or "*SET* PDU" to another node. We define this type of management operation as normal management operation.

Sending management information periodically to the cluster members is cost ineffective that implies too many management packets cause high message overhead and bandwidth wastes. Need-to-know basis management allows the efficient use of the resources. In this management, when manager wants the information, it can request the agent to send specific management attributes that are needed to know for some management applications.

4. DMMS FUNCTIONS

The use of distributed network management with MIB elements for control and monitoring is extremely cost effective compared to conventional network management methods where expensive devices are used to monitor the network at different locations of the network globally. In our proposed distributed network management approach, each source-destination pair controls their applications independently without requiring any knowledge to be accessed from the network itself. In this section, we have shown some management applications that can be used for controlling and monitoring purpose.

4.1. Real Time Network Control Functions

Real Time Network Control Function allows understanding the computational measurements for end user applications. The routing type of application at the IP layer which sets up the paths across the network requires knowledge of nodes in the vicinity and their traffic conditions. The Position, Location and Tracking (PL&T) [11] algorithm allows real time computation of PL&T data for each moving target node at different instances of time which is sent to the Cluster Head that maps the network and broadcast the mapping data of all nodes, so that each node can use the data for any end user application. However, the user protocol layers may have to send this data in terms of strings of X and Y coordinates using floating point function which occupies a lot of bandwidth, while exchanging the MIB elements reduces the bandwidth requirement significantly at the cost of computation of Cluster map at different nodes individually is increased. However, if each node computes only the need-to-know information such as its neighbors, the computational burden can be significantly reduced. These computations need to be done in real time as the cluster mapping changes periodically as nodes move.

4.1.1. Management Aspect on Position, Location & Tracking (PL&T) Application

Position, Location & Tracking (PL&T) Application is a user application that is used to determine the location of a target node at any instance of time through triangulation and zone finding process in a network [11]. In [11], using PL&T method, in a cluster, a reference node can determine the X_Y coordinate of a location point for any target node. After finding the coordinates of a target node, it will send all the X_Y strings to the Cluster Head. Then, Cluster Head broadcast these strings of X_Y coordinates to all other nodes. On the other hand, if we apply DMMS to this PL&T application, reference node need to create some management attributes or MIB elements that are needed to determine the X_Y coordinates of any target node. After creating the management attributes, reference node will send all these management attributes to the Cluster Head. After receiving these management attributes, Cluster Head itself can use it for determining the coordinates of any target node or CH can broadcast all these management attributes to all other nodes in the cluster. By using these management attributes, any node can determine the X_Y coordinate of any other node according to its need.

For simulation, we chose the management packets to carry Position, Location & Tracking (PL&T) data as management attributes. The management attributes contain the PL&T data that include the target id and the PL&T. The cluster head collects the PL&T for each node from the reference nodes used in triangulation. The PL&T algorithm proposed in [11] was expected to send the PL&T of each node to the cluster head. The cluster head organizes the PL&T data in terms of management attributes where a management object (PLT_MO) has multiple attributes with node_ID, PL&T Data (X, Y) and time when computed (T). In Figure 5, through the sequence diagram, we have presented how management attributes can be set for PL&T. The network management packets from each ensemble are used to send a collection of these attributes in a broadcast mode so that every node in the cluster would be able to develop a map of the cluster on the screen.

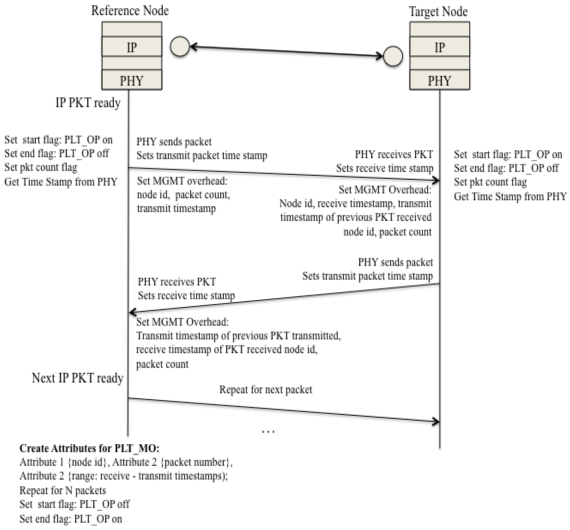

Figure 5. Sequence diagram of creating management attributes for PL&T

In user application of PL&T [11], for sending PL&T information from reference node to the Cluster Head, a large number of strings need to be transmitted that requires large amount of bandwidth. Whereas, apply DMMS in PL&T, for sending PL&T information from reference node to the Cluster Head, only related management attributes need to be transmitted that requires comparatively less bandwidth. Thus, applying DMMS, a PL&T method can be more bandwidth efficient.

On other aspect, Applying DMMS on PL&T, it can provide the flexibility on using PL&T information. Since for determining the PL&T, the management attributes of all the nodes are available to every node, any time a node can use those attributes to determine the PL&T of any other nodes according to its need. Nodes don't need to calculate the PL&T for all the cluster members all the times. In the next section, through simulation result, we will show that, the message overhead of user application of PL&T will be less if we apply DMMS on PL&T application.

4.1.2. Management Aspect on Graph Theoretical Routing Application

For controlling purpose, DMMS can be used on different MANET routing applications. In MANET environment, nodes move randomly. Therefore, to get the position of the nodes at any instance of time is an important attribute for any MANET routing algorithm. In cluster based MANET, cluster head (CH) have many responsibilities such as to manage the resources of the cluster, to maintain the communication between two clusters, and etc. By applying DMMS on one of the graph theoretical routing protocols [12], we have discussed how DMMS can be used in controlling purpose to manage message overhead, to offer flexibility, and to provide resource efficiency. In graph theoretical routing [12], PL&T data is required to develop a snap shot for the cluster. By using user application of PL&T [11], reference nodes periodically find the PL&T data in terms of X-Y coordinates of all the nodes and sends those X_Y strings to the CH for developing a snap shot of the cluster. Then, CH periodically broadcast the processed snap shot to all the cluster members for updating the PL&T information. On the other hand, if we apply DMMS on PL&T application for developing the snapshot of the cluster, it can reduce the message overhead and provide flexibility in similar way that we have discussed in previous sub section. Furthermore, by using DMMS, it is also possible to save some processing power of each node. Since all the management attributes of PL&T of all nodes are available to each node, to update the PL&T of any node at any instance of time, a node can calculate the PL&T of its neighbor nodes by using the management attributes and PL&T information of those nodes that was calculated previously.

4.2. Network Monitoring Function

Network monitoring involves developing the network behavior at every node to derive the performance in terms of throughput (IP layer), delay and data rate (physical layer). The proposed DMMS system involves maintaining a network management agent in each node with an associated Management Information Base (MIB). Also, the system will provide the ability of any management function in a node to derive the performance at any other remote node only through the knowledge of MIB elements which is set up in terms of Managed Object (MO) and its associate Attributes. This is achieved by accessing the MOs and their associated attributes to compute the performance for monitoring purposes. The fact that only MIB elements are accessed will enable efficient use of bandwidth for management function. Traditional networks use spoofing type of equipment that tends to be expensive.

At each node, the management function is set up as an “agent process” with a set of MIB elements. For each user protocol layer, there is a Managed Object (MO) and a set of associated Attributes. The Attributes are updated locally through a Layer Management Entity (LME) for any behavioral change at each user protocol layer. The attributes are set up in terms of “counters”, “timers”, “status flags” and “thresholds”. The content of the attributes are updated at each protocol layer whenever there is a behavioral change occurs. For an IP layer protocol, every packet transmitted at the sending node sets a flag at the LME and the content of a counter (used as Attribute) is updated. Similarly, the data rate counter at the Physical layer can be updated also.

4.2.1. User Data Throughput Computation

To compute the user data throughput of message, packet, and frame for a cluster, we have to use Application layer MO (APP_MO), IP layer MO (IP_MO) and MAC layer MO (MAC_MO) respectively where all the attributes of these MOs will be defined in terms of counters, thresholds, timer and flags. The detailed steps to compute the packet throughput given below:

<u>Setting IP layer Manage Object (MO):</u>

```
IP_MO
{
      sent_packet_counter (integer, value1)
      start_sent_packet_timer_count (time, value2)
      start_sent_packet_timer_flag (boolean, value3 := 0 or 1)
      sent_threshold_packet_count (integer, value4 = N)
      received_packet_counter (integer, value5)
      start_received_packet_timer_count (time, value6)
      start_received_packet_timer_flag (boolean, value7 := 0 or 1)
      received_threshold_packet_count (integer, value8 = N)
}
```

<u>Initialized PHY_MO:</u>

Before the transmission start, management attributes such as “sent packet counter”, “start sent packet timer”, “received packet counter” and “start received packet timer” of IP_MO need to be initialized.

<u>Management Operations:</u>

When IP packet is ready to send, IP layer sets the sent timer flag equals to true (*start_sent_packet_timer_flag = 1*). If sent flag equals to true or 1, sent packet timer (*start_sent_packet_timer_count*) starts the timer and sent packet counter (*sent_packet_counter*) starts counting number of packets. After sending each IP packet, sent packet counter and sent packet timer will be updated. After transmitting the first ensemble of packets, management attributes (*sent_packet_counter*, *start_sent_packet_timer_count*) of first ensemble is sent through management packets of second ensemble (consist of both management packets and user data packets) from the sending node to the receiving node. These management packets are used as a SET type management packet.

In the receiving node, when node starts receiving first ensemble of packets, IP layer sets the received timer flag equals to true (*start_received_packet_timer_flag = 1*). If received flag equals to true or 1, received packet timer (*start_received_packet_timer_count*) starts the timer and received packet counter (*received_packet_counter*) starts counting number of packets. After receiving the first ensemble of packets, management attributes (*received_packet_counter*, *start_received_packet_timer_count*) of that ensemble are sent through SET type management packets of the second ensemble from the receiving node to the sending node.

For each transmission of IP packet in an ensemble, the content of the "*sent_packet_counter*" is increased and management functions check whether it reaches the value specified in the "sent_threshold_packet_count" at sending node. Similarly, the content of the "*received_packet_counter*" is increased and management functions can check whether it reaches a value specified in the "received_threshold_packet_count" for each packet received at receiving node. When the transmission of second ensemble of packets begins, the attributes with its contents updated during the transmission of the first ensemble are stored in the buffer for computation of the monitoring values. Since IP packets are continuously transmitted, management attributes are not initialized after transmitting of each ensemble. Therefore, the computation of the next monitoring value takes into first and the second ensemble. Similarly, as multiple ensembles are transmitted, the computation is cumulative.

Both sending node and receiving node have the management attributes (*sent_packet_counter*, *start_sent_packet_timer_count, received_packet_counter*, *start_received_packet_timer_count*).
After each ensemble is exchanged between two nodes, the performance management function calculates the throughput by using the following equations (1 - 3).

$$time\ to\ sent_{n\ ensambles} = time\ stamps\ end\ sent_{first} - \ time\ stamps\ start\ sent_{n} \ldots\ldots\ldots (1)$$

$$time\ to\ received_{n\ ensambles} = time\ stamps\ end\ received_{first} - \ time\ stamps\ start\ received_{n} .. (2)$$

$$Cumulative\ Throughput_{n} = \frac{\frac{number\ of\ packets\ received_{n}}{time\ to\ receive_{n}}}{\frac{number\ of\ packet\ sent_{n}}{time\ to\ send_{n}}} \ldots\ldots\ldots (3)$$

Average throughput of all the cumulative throughputs has found through equation (4).

$$Avg.Throughput = \frac{\sum_{n=1}^{number\ of\ ensembles} Cumulative\ Throughput_{n}}{number\ of\ ensembles} \ldots\ldots\ldots (4)$$

Application layer MO, MAC layer MO and IP layer MO are used to find out the message, frame, and packet related network performance respectively.
Through the above procedure, we have shown, to find the throughput we have setup the MIB elements in terms of counter, flag and threshold into every nodes in a cluster and computed the throughput without requiring any expensive equipment. In the next section, we have simulated an example to find the throughput. We have also discussed below about what are the effects on network behaves if we find the throughput using the MOs.

4.2.2. Delay Computation

To measure the end-to-end packet propagation delay and each ensemble time delay for transmitting of IP packets between two nodes, one of the ways we can determine is based on using *Time of Departure (ToD)* and *Time of Arrival (ToA)* for each packet and another way is long term average method. Both of the techniques of finding packet delay have shown below.

<u>ToD and ToA based management computation for time delay:</u>

The ToD and ToA computation is done at the physical layer of the OSI reference model to avoid the processing delays at higher layers. The processing delays at the IP and MAC layers can vary statistically based on the operating system function calls. At the physical layer, as the packet leaves the port, it is easier to time stamp and similarly, at the receiver, time stamping is done as soon as the packet is received at the physical port.

The management packet sends all ToDs corresponding to packet IDs to the receiving side and the receiving side sends all ToAs to the sending side of the packets with IDs received in each ensemble. Similarly, for full duplex operation, the reverse management function does the same. Thus, each management packet from both sides will have ToDs and ToAs. This will allow each node to compute the delays independently. We have shown the Physical Layer Manage Object (PHY_MO1) in below which is used for ToD and ToA based management computation for propagation delay.

```
PHY_MO2
{
      packet_id(integer, value11)
      time_of_departure(time, value12)
      time_of_arrival(time, value13)
      sent_packet_time (packet_id,time_of_departure)(value 11,value12)
      received_packet_time(packet_id,time_of_arrival)(value 11,
                                                       value13)
}
```

Figure 6 shows the sequence diagram that presents the procedure of how we have created management attribute for physical layer that are used for finding the propagation delays.

$$Packet\ time\ delay, d_i = ToA_i - ToD_i;\ for\ each\ packet\ packetID_i\ \ldots \ldots \ldots (5)$$

Where, ToD_i is recorded at the sending side and ToA_i is recorded at the receiving side.

$$Average\ Packet\ time\ delay\ = \frac{\sum_{i=0}^{N} d_i}{N} \ldots \ldots \ldots (6)$$

where, N = number of packets in a time

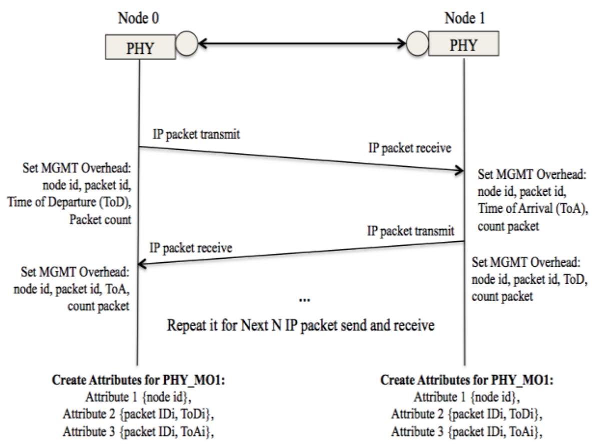

Figure 6. Sequence diagram of creating the management attributes for physical layer MO (PHY_MO) for finding the time delay

Long Term Average based management computation for time delay:

The second method is to compute for a period of time, how many packets are transmitted from the sending node and compute how much time it takes to receive all these packets at the receiving node. In this method, for a reasonable number of packets in an ensemble, the time criticality is reduced compared to the *ToD* and *ToA* measurements. We have shown the Physical Layer Manage Object (PHY_MO2) in below which is used for long term average based management computation for time delay.

```
PHY_MO2
{
      node_id
      sent_ensemble_start_time (time, value9)
      received_ensemble_end_time (time, value10)
}
```

We have sent N packets and measured the time it took for sending N packets. On the other hand, we have also received N packets and measured the time it took to receive N packets. For measuring the total time it took for sending N packets, we have kept records the time of departure for the 1^{st} and n^{th} packet of every ensemble in sending side. Again, for measuring the total time it took for receiving N packets, we have kept records the time of arrival for the 1^{st} and n^{th} packet of every ensemble in receiving end. In the receiving end, we have calculated the delay by using the sending side information that is sent through the management packet after each ensemble. The total delay is the difference between the amount of time it took to receive the packets at the receiving side and the amount of time it took to send the packets at the sending side. This method to find the end-to-end delay is called *long-term average method*. Since this is a long term average, it is more consistent than the ToD, ToA.

The time delay for each ensemble is the difference of the time stamps sent from the source and received from the source that is founded by using equation (7). After finding the time delay for each ensemble, average delay is founded by using equation (8).

$$delay_i = time\ stamps\ end\ receiving_i - \ time\ stamps\ start\ sending_i \ldots\ldots\ldots (7)$$

$$Avg.delay = \frac{\sum_{i=1}^{number\ of\ ensembles} delay_i}{number\ of\ ensembles} \ldots\ldots\ldots (8)$$

4. SIMULATION RESULTS

The proposed DMMS was implemented in real time using Network Simulator 2 (NS2). The performance results of the simulation are plotted using the MATLAB software. In the simulation, we used a MANET cluster of size 100 x 100 sq. m. area. In the cluster, we placed initially 10 nodes randomly at different locations as shown in Figure 7 and then varied the number of nodes up to 100 in steps of 10 to study the performance. The simulation parameters have given in Table 2.

Table 2: Simulation parameters

Parameter	Values
Simulation time	1000 seconds
Area	100m x 100m
Number of Nodes	10, 20, 30, 40, 50, 60, 70, 80, 90, 100
Node Placement Strategy	Random
Mobility of Nodes	Random Min speed: 1.00 m/s Max speed: 5.00 m/s Avg. speed: 2.49 m/s
Capacity of the Network	20 Mbps
Packet size	512 bytes
Ensemble Size (sent each time from a node to another)	100 packets (Fixed Size)
Number of Management packet each ensemble	2 packets (Initially) and is variable. (Remaining for user packets)

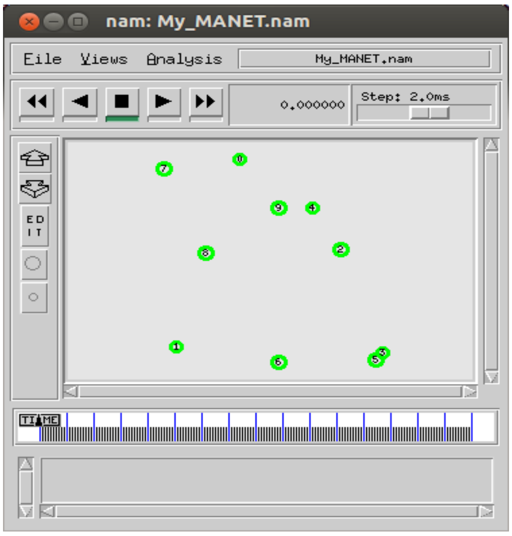

Figure 7. A cluster with the random distribution of nodes

5.1. Simulation Results for Monitoring Function

In this simulation, we formed a cluster initially with 10 nodes randomly placed within the cluster. The simulation allows every node to communicate with every other node. For monitoring function, the simulation has been set up to have pair wise communication where communicating node pairs are: *node 0 and node 1, node 2 and node 3, node 4 and node 5, node 6 and node 7, and node 8 and node 9*

Also, the nodes can start sending packets at any instance of time during the simulation. We have used Constant Bit Rate (CBR) application with a rate of 200 Kbits/sec for transmitting IP packets where each packet consists of 512 bytes. We have declared and initialized specific management attributes for some of the MOs of OSI Layer that are used for computing the network performance. We have considered the node pairs of 0 and 1 for computation. When *node 0* starts transmitting an ensemble of data packets to *node 1*.

5.1.1. Throughput Computation

For each transmission of IP packet in an ensemble, we increase the content of the "*sent_packet_counter*" and check whether it reaches the value specified in the "sent_threshold_packet_count" at node 0 (sending node). We increase the content of the "*received_packet_counter*" and check whether it reaches a value specified in the "received_threshold_packet_count" for each packet reception at node 1 (receiving node). When the transmission of second ensemble of packets begins, the attributes with its contents updated during the transmission of the first ensemble are stored in the buffer for computation of the monitoring values. Since IP packets are continuously transmitted, management attributes are not initialized after transmitting of each ensemble. Therefore, the computation of the next monitoring value takes into first and the second ensemble. Similarly, as multiple ensembles are transmitted, the computation is cumulative.

Both sending node and receiving node have the management attributes. After each ensemble is exchanged between two nodes, the performance management function calculates the throughput.

Table 3. Management attributes of first 10 ensembles between node 0 and node 1 for finding the user data throughput

Number of Ensembles	Number of packet sent	Number of packet received	Time (sec) to sent by node 0	Time (sec) to received by node 1
1	100	100	2.02752	2.02786
1,2	200	200	4.05504	4.05558
1,2,3	300	300	6.08256	6.08384
1,2,3,4	400	400	8.11008	8.1115
1,2,3,4,5	500	500	10.1376	10.14122
1,2,3,4,5,6	600	600	12.16512	12.16944
1,2,3,4,5,6,7	700	700	14.19264	14.19794
1,2,3,4,5,6,7,8	800	800	16.22016	16.22592
1,2,3,4,5,6,7,8,9	900	900	18.24768	18.25652
1,2,3,4,5,6,7,8,9,10	1000	1000	20.2752	20.28716

Table 3 shows the management attributes for the transmission of the first 10 ensembles of packets between node 0 and node 1 that we have found from the real time simulation result and stored in the buffer of node 0 and node 1. Same way we have also found the management attributes for other 4 pair of nodes.

Thus, both *node 0* and *node 1* have all the management attributes that are listed in Table 3. The ratio of the number of packets received at the destination and the number of packets transmitted from the source within the ensemble time will give the throughput, which can be measured independently by performance management function of the sending node and the receiving node. In Table 4, we have shown the throughput of each ensemble based on the simulation data.

By observing all the data throughputs, cumulative user data throughputs is not 100% because of some delay occurred to receive the ensembles. By averaging cumulative throughputs for all ensembles we have found the average throughput for link 1 (node 0 to node 1). Following the same procedure we have also found the average throughputs for other pairs of nodes. In Table 5 we have given the average throughput for all pair of nodes. By averaging five pair of nodes throughput we have found the average throughput for the entire cluster.

Table 4. User data throughput of first 10 ensemble of packet for node 0 to node 1

Number of ensembles	**Data Throughput**
1	0.999834
1,2	0.999901
1,2,3	0.999635
1,2,3,4	0.999931
1,2,3,4,5	0.998916
1,2,3,4,5,6	0.999654
1,2,3,4,5,6,7	0.999517
1,2,3,4,5,6,7,8	0.999773
1,2,3,4,5,6,7,8,9	0.998483
1,2,3,4,5,6,7,8,9,10	0.998463

Table 5. Average user data throughputs for all pair of nodes in a cluster

Nodes Pair	**Average Throughput**
node 0 to node 1	0.999703
node 2 to node 3	0.999665
node 4 to node 5	0.999778
node 6 to node 7	0.999726
node 8 to node 9	0.999883

From simulation result, we have found the throughput of the ensembles for node 0 to node 1 and other simulation parameters related to the cluster has set as mentioned in the Table 2. Figure 8 shows the graph of the throughput for 200 ensembles of data packets where X-axis presents the serial of the ensembles and Y-axis presents the throughput of those ensembles.

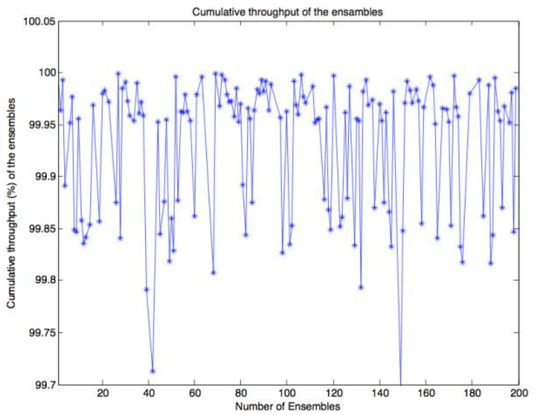

Figure 8. Throughput calculations of a link for each ensemble of data packets by exchanging IP_MO

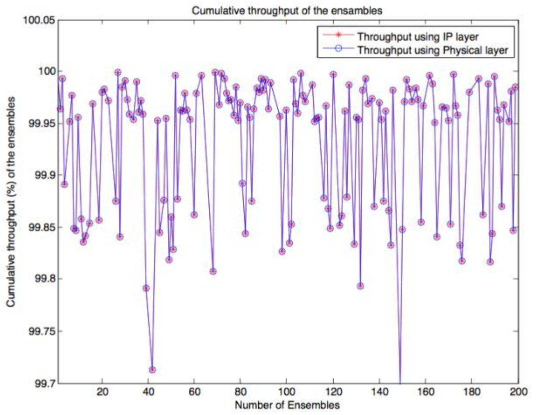

Figure 9. Verification of finding throughput in two different layers

We have verified the accuracy for finding throughput by calculating the throughput in both IP layer and physical layer. For physical layer we have counted the total number of bytes that are sent from node 0. Also we have counted the total number of bytes that are received at node 1. Again we have counted the time for sending and receiving. However, we did not consider the mac frame for computing the cumulative throughput in physical layer. Form Figure 9, we have seen the throughput of a link (node 0 to node 1) that we have calculated by using IP layer MO and physical layer MO are completely matched with each other.

5.1.2. Delay Computation

By using simulation results, we have shown the detail computation for finding delays for node 0 to node 1. In simulation, node 0 has transmitted an ensemble of data packets to node 1, the management agent of node 0 has sent a management packet which is the first packet in that ensemble of data packets. The payload of the management packet consists of different management attributes such as "Time of Departure (ToD)" of all the packets in the ensemble of the sending node 0 and the "Time of Arrival" (ToA) of the received packets from receiving node 1. In the reverse channel from node 1 to node 0, the ensemble contains ToD of the packets sent by node 1 in the ensemble and ToA of the packets received from 1. Thus, it allows both node 0 and node 1 to have ToD and ToA of all the packets sent and received. The difference between ToA and ToD has provided the propagation time between two nodes. Based on the simulation result, Table 6 presents that list of management attributes for first 10 IP packets transmission that we have used for finding delay.

Table 6. List of management attributes for first 10 IP packets that we have used for finding delay

Packet number	**Packet ID**	**ToD**	**ToA**
1	5	1.00645	1.0109
2	6	1.02693	1.02971
3	7	1.04741	1.05055
4	8	1.06789	1.07057
5	9	1.08837	1.09143
6	10	1.10885	1.11159
7	11	1.12933	1.13237
8	12	1.14981	1.15259
9	13	1.17029	1.17301
10	14	1.19077	1.19341

Table 7. Calculated propagation delays and average propagation delays for first 10 packets of an ensemble

Packet number	**Packet ID**	**Delay (sec)**	**Avg. delay (sec)**
1	5	0.00444336	0.00444336
2	6	0.00277612	0.00360974
3	7	0.00313612	0.00345187
4	8	0.00267612	0.00325793
5	9	0.00305612	0.00321757
6	10	0.00273612	0.00313733
7	11	0.00303612	0.00312287
8	12	0.00277612	0.00307952
9	13	0.00271612	0.00303915
10	14	0.00263612	0.00299884

We have calculated the delay for each packet transmission and have calculated the average delay for an ensemble of packets by using equation 5 and 6.

Table 7 presents all the delays that performance management function has calculated. Based on the simulation results, Figure 10 shows the graph of the delay for each packet transmission of an ensemble of packets and Figure 11 shows the graph of average delay for that ensemble of packets. For performing this simulation, the parameters that we have used are listed in Table 2.

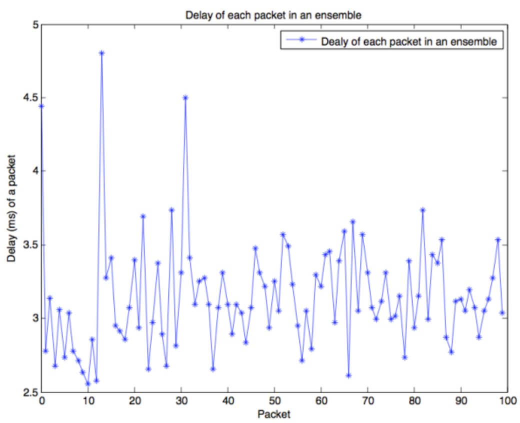

Figure 10. Propagation delay of each packet of an ensemble by using PHY_MO2

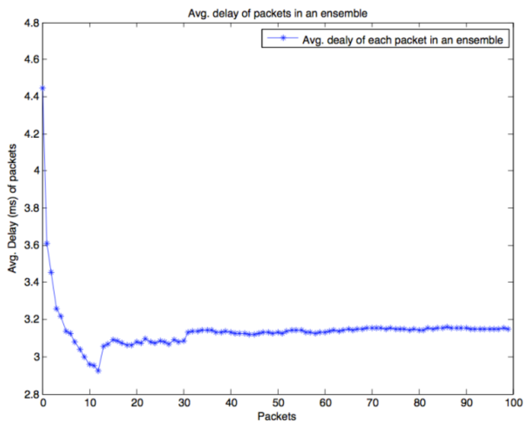

Figure 11. Average propagation delay for an ensemble of packets by using PHY_MO2

In another method, we have recorded the time when the transmission of an ensemble of packets start and have record of the time when the transmission of an ensemble of packets end. Using this two-parameter, we have found the time delay of each packet transmission of an ensemble of packets from node 0 to node 1 and the average the time delay for that ensemble of packets from node 0 to node 1.

Node 0 has transmitted an ensemble of data packets to node 1. When the first packet is transmitted, node 0 has recoded the sent_ensamble_start_time. At node 1, it also recoded the received_ensamble_end_time when the transmission of an ensemble of packets ends. In the reverse channel from node 1 to node 0, the ensemble contains the received_ensamble_end_time management attribute for the previous ensemble. Thus, it allows both node 0 and node 1 to have sent_ensamble_start_time and received_ensamble_end_time of the ensemble sent and received. The difference between these two management attributes has provided the time delay of an ensemble between node 0 and node 1.

Table 8 shows the management attributes for finding the time delay using long term average method for the transmission of first 10 ensembles of packets between node 0 and node 1 that we have found from the real time simulation result.

Table 8. List of management attributes for first 10 ensembles that has used for finding the time delay

Ensemble Number	sent_ensamble_start_time (sec)	received_ensamble_end_time (sec)
0	1.02693	3.05757
1	3.07493	5.10581
2	5.12293	7.15323
3	7.17093	9.20191
4	9.21893	11.2523
6	13.3149	15.3485
7	15.3629	17.3936
8	17.4109	19.4448
9	19.4589	21.4925
10	21.5069	23.5374

Table 9. Calculated time delays and average delays for first 10 ensembles

Ensemble Number	Time delay (sec)	Average time delay (sec)
1	2.03063	2.03063
2	2.03087	2.03075
3	2.03029	2.030596667
4	2.03097	2.03069
5	2.03337	2.031226
6	2.03361	2.031623333
7	2.03069	2.03149
8	2.03391	2.0317925
9	2.03355	2.031987778
10	2.03049	2.031838

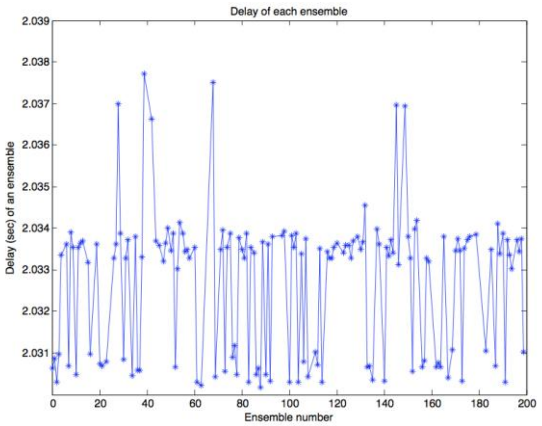

Figure 12. Delay of each ensemble by using long-term average method using PHY_MO2

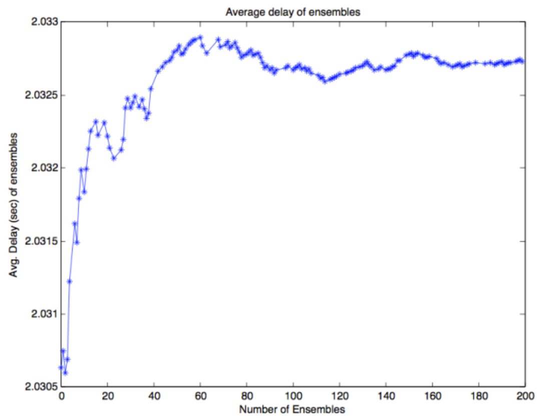

Figure 13. Average delays for 200 ensembles of packets by using long-term average method using PHY_MO2

Figure 12 shows the graph of the delay for each ensemble of packets where each ensemble consists of 100 packets. Figure 13 shows the graph of average delay for 200 ensembles of packets where the average delay is calculated after end of each ensemble.

We find the average time delay for each packet by ratio of total time delay for an ensemble and number of packets in an ensemble. However, this average time delay for each packet is not the same as propagation delay because in the time delay we also consider the packets generation time. In our simulation, we have considered data rate of 200 kbps and packet size is 512 bytes.

5.2. Simulation Results for Monitoring Function

We have also run the simulation to analyze the performances of DMMS for controlling purpose on a particular user application of PL&T. In user application of PL&T [11], for sending PL&T information from reference node to the Cluster Head, a large number of strings need to be transmitted that requires large amount of bandwidth. Whereas, apply DMMS in PL&T, for sending PL&T information from reference node to the Cluster Head, only related management attributes need to be transmitted that requires comparatively less bandwidth. Thus, applying DMMS, a PL&T method can be more bandwidth efficient. Figure 14 shows the comparison between the data (bytes) need to transmit for performing PL&T using user application, PL&T using management attributes for different size of the clusters and PL&T using management attributes for 15% of nodes for different size of the cluster. From Figure 14, through simulation results, we have shown that the message overhead of user application of PL&T is higher than the message overhead of PL&T application that uses DMMS. In Table 10, we have given the attributes list those are used to simulate the message overhead of PL&T.

Table 10. Management attributes and user application data list used for one node PL&T

User application Attributes	**Size**	**Management attributes**	**Size**
X coordinate	32 bytes	3 Ranges (ToA - ToD)	48 bytes
Y coordinate	32 bytes	Node id	2 bytes
Node id	2 bytes		

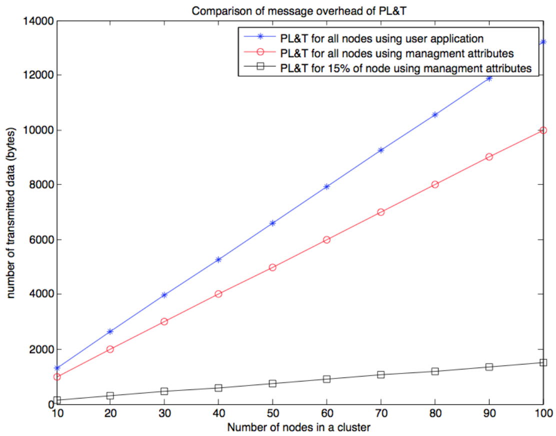

Figure 14. Message overheads of user application of PL&T and PL&T application that uses DMMS.

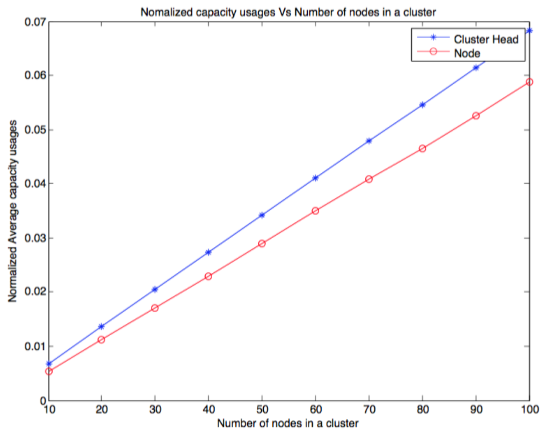

Figure 15. Normalized capacity usages for CH and average capacity usages of all the nodes

5.3 Performance Analysis of DMMS

To analyze the performance of DMMS, we have find out the capacity usage of a cluster. Here, we have determined capacity usage of the CH and average capacity usage of all the nodes when CH needs to know the throughputs of all the nodes in the cluster. CH broadcasted a request for sending the management attributes of IP layer MO that are node id (1 bytes), 4 attributes id (1+1+1+1 or 4 bytes), total packets sent and received (2+2 or 4 bytes), and total time takes to send and receive those packets (8+8 or 16 bytes). Total attributes size for one node is 25 bytes. In our simulation, we assume that CH has 30% network capacity and rest of the node has 70% network capacity. We use 20Mbps of total capacity, CBR application of 200 kbps data rate and packet size is 512 bytes. If number of node increases, CH capacity remains the same but nodes capacity decreasing. According to this request, all the nodes sequentially sent the listed management attributes to the CH through a management packet that will interleave with an ensemble of user data packets. After getting the management attributes, CH found the throughputs of all the nodes in the cluster. In simulation, we have varied the number of nodes from 10 to 100 for a cluster to find out the capacity usage. We are considering only the management packets sent and received for finding the capacity usages.

When the number of nodes in a cluster increases, the maximum capacity of each node decreases. Again, normalized capacity usage is the ratio of capacity usage and maximum capacity. Thus, if the maximum capacity of each node decreases, it increases normalized capacity of all the nodes. In Figure 15, we have seen that if we increase the number of nodes in a cluster, it increases the capacity usage for management packet linearly.

We have also analyzed the impact of delay for finding the throughputs of all the nodes in a cluster. CH cannot receive the management packet from all nodes at a time, since management packet of each node is sent through several ensembles. Thus it causes delay to calculate the throughput of all the nodes. Therefore, if the number of node in a cluster increases, it increases the delay accordingly.

We interleave both management packets and user data packets. We create an ensemble of packets to include the first two packets containing management information (Attributes and Managed Objects) and the remaining packets correspond to the user data packets. Since the number of management packets is significantly smaller within each ensemble compared to the user data packets, the throughput for the management information exchange is only a fraction of the throughput for data packets. It does not mean that the throughput is low for management packets. If we rewrite the argument using the bandwidth concept, where the management is only allocated 2% of the total bandwidth while the user data packets are allocated 98% bandwidth (ignoring the packet overhead), 2% is completely used for management and therefore, the throughput for management is 100%. That is, this scenario identifies that you only need two management packets per ensemble for management. This is also true whether the management packets are inter-leaved or sent in a dedicated channel of a finite bandwidth for management. However, if the network needs more management packets to be sent, then a dedicated channel would be more efficient.

6. CONCLUSIONS

In this paper, we introduced resource efficient Distributed MANET Management System (DMMS), which uses fully distributed management architecture. The general manger and the “single point of failure” problems have solved by using DMMS. We developed a new set of Management Information Base (MIB) elements or management attributes and stored the resource activities in terms of counters, timer, status flag and threshold values. We have shown that the abstract data is exchanged between different management agents residing in different resources on a need-to-know basis and each agent logically executes management functions locally to develop understanding of the behavior of all network resources for monitoring purposes (finding throughput, delay) and controlling purposes (PL&T application, managing routing algorithm, etc.). In this paper, we presented the MANET architecture with MIB based distributed network management and demonstrated bandwidth efficiency for management operations that include monitoring of resources, and performance management for multi-service provisioning. We have also analyzed the performance of DMMS to show the efficient resource usages.

ACKNOWLEDGEMENTS

This work is supported in part by the US Army Research Office under the Research Cooperative Agreement Grant, W911NF-04-2-0054 to the ARO Center for Battlefield Communications (CeBCom), National Science Foundation under the contract 0931679 and 1229744, Department of Electrical and Computer Engineering, Prairie View A&M University (Texas A&M University System). The views and conclusions contained in this dissertation are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government.

REFERENCES

[1] A. Leinwand, K. Fang, Network Management: A Practical Perspective, Reading, MA: Addison-Wesley, 1993.
[2] W. Stallings, SNMP, SNMPv2, and CMIP: The Practical Guide to Network Management Standards, 1st edition. Reading, MA: Addison-Wesley, 1993.
[3] Chen, W., N. Jain, and S. Singh. 1999. “ANMP: Ad Hoc Network Management Protocol,” IEEE J. on Select Areas Communication, Vol 1, No. 8, pp. 1506-1531.
[4] C. Tuduce, T. Gross, “Resource Monitoring Issues in Ad Hoc Networks,” International Workshop on Wireless Ad-Hoc Networks, IEEE, pp. 259 – 264, Jun. 2004.
[5] Z. Zhao, W. Huangfu, Y. Liu, L. Sun, “Design and Implementation of Network Management System for Large-scale Wireless Sensor Networks,” International Conference on Mobile Ad-hoc and Sensor Networks, Beijing, pp. 130-137, 2011.
[6] J. Kim, H. Jeon, J. Lee, “Network Management Framework for Wireless Sensor Networks (WSN),” Proceedings of Held as Part of the Future Generation Information Technology Conference (FGIT), Part I, 2010.
[7] C. Shen, C. Jaikaeo, C. Srisathapornphat, Z. Huang, “The Guerrilla Management Architecture for Ad hoc Networks,” in Proc. of MILCOM, Vol. 1, 2002.
[8] Ayari, M., Kamoun, F., Pujolle, G. 2008, “Towards Autonomous Mobile Ad Hoc Networks: A Distributed Policy-Based Management Approach,” Wireless and Mobile Communications, 2008. ICWMC '08, Page(s): 201 – 206
[9] L.B. Ruiz, J.M. Nogueira, and A.A.F. Loureiro, “MANNA: A Management Architecture forWireless Sensor Networks,” IEEE Communications Magazine, vol. 41, no. 2, pp. 116–125, 2003.
[10] G.R. Khan, D.R. Vaman, “Modeling and Designing Resource Efficient Distributed MANET Management System (DMMS)”, International journal of Computer Networks & Communications, Vol.6, No.5, Sep. 2014.
[11] K. Patel, D.R. Vaman, “Graph Theoretic Routing Algorithm (GTRA) For Mobile Ad-Hoc Networks (MANET)”, International Journal on Applications of Graph Theory in Wireless Ad hoc Networks and Sensor Networks (GRAPH-HOC), Vol 5, No. 4, Dec. 2013
[12] S. Khan, D. R. Vaman, S. T. Koay, “Highly Reliable Multi-Service Provisioning Using Sequential Prediction of Zone and Pl&T of Nodes In Mobile Networks”, International Journal of Wireless & Mobile Networks (IJWMN), vol. 6, No. 4, August, 2014

Authors

Golam R. Khan is currently Ph.D candidate in Electrical Engineering from Prairie View A&M University (PVAMU), TX and working as a Graduate Teaching Assistant in the same Department. Previously, he was working as a Graduate Research Assistant in the Center of Battlefield Communication (CeBCom). His advisors are Prof. Dhadesugoor R. Vaman, and Dr. Suxia Cui. His current research interest is "Modeling and designing resource efficient Distributed MANET Management System (DMMS) for multi-service applications". He got his Bachelor of Science degree in Computer Science (CS) from American International University Bangladesh (AIUB), Bangladesh in May 2006. After his B.Sc, he worked in industries for 2 years. He also completed his Master's degree in Electrical Engineering (EE) from Tuskegee University (TU), AL in December 2010.

Sharmistha Khan got the B.Sc. degree in Computer Science from American International University-Bangladesh (AIUB), Dhaka, Bangladesh in 2006, and the M.S. degree in Electrical Engineering from Tuskegee University, Tuskegee, AL in 2011, respectively. She is currently pursuing her PhD degree in the department of Electrical and Computer Engineering at the Prairie View A & M University, Prairie View, TX. She is working under the supervision of Prof. Dhadesugoor R. Vaman and Prof. Siew T. Koay at Prairie View A & M University. Her research area includes Mobile Adhoc Network, Cognitive Radio Networks, Sensor Networks, Mobile WiMAX Technology, Handoff Management, and Handoff Performance and Decision Making Algorithms for Broadband Wireless Networks.

Dhadesugoor R. Vaman was Texas Instrument Endowed Chair Professor and Founding Director of ARO Center for Battlefield Communications (CeBCom) Research, ECE Department, Prairie View A&M University (PVAMU). He has more than 38 years of research experience in telecommunications and networking area. Prior to joining PVAMU, Dr. Vaman was the CEO of Megaxess (now restructured as MXC) which developed a business ISP product to offer differentiated QoS assured multiservices with dynamic bandwidth management and successfully deployed in several ISPs. Prior to being a CEO, Dr. Vaman was a Professor of EECS and founding Director of Advanced Telecommunications Institute, Stevens Institute of Technology (1984-1998); Member, Technology Staff in COMSAT (Currently Lockheed Martin) Laboratories (1981-84) and Network Analysis Corporation (CONTEL)(1979-81); Research Associate in Communications Laboratory, The City College of New York (1974-79); and Systems Engineer in Space Applications Center (Indian Space Research Organization) (1971-1974). He was also the Chairman of IEEE 802.9 ISLAN Standards Committee and made numerous technical contributions and produced 4 standards. Dr. Vaman has published over 200 papers in journals and conferences; widely lectured nationally and internationally; has been a key note speaker in many IEEE and other conferences, and industry forums.

Suxia Cui is an associate professor in the Department of Electrical and Computer Engineering at Prairie View A&M University (PVAMU). She received her BS and MS degrees both in Electrical Engineering from Beijing University of Technology in 1996 and 1999 respectively. Right after obtained her PhD degree in Computer Engineering from Mississippi State University in 2003, Dr. Cui joined Prairie View A&M University. Her research interests include image and video processing, data compression, wavelets, computer vision, remote sensing, and computing education. She has published numerous papers, and her research is funded by United States Department of Agriculture, Department of Education and National Science Foundation.