IOSR Journal of Engineering (IOSRJEN) **www.iosrjen.org**
ISSN (e): 2250-3021, ISSN (p): 2278-8719
Vol. 08, Issue 5 (May. 2018), ||VI|| PP 69-77

Performance Analysis of Routing Protocols

N Dinesh Kumar[1], V.S K Reddy[2]
[1]*Research Scholar (Electronics & Communication Engineering), Rayalaseema University, Kurnool, Andhra Pradesh, India*
[2]*Malla Reddy College of Engineering & Technology, Kompally, Hyderabad, Telangana State, India*
Corresponding Author: N Dinesh Kumar

Abstract: Networking plays a prominent role in present day scenario because of its vast expansion of technology throughout the world. However the purpose of networking still remains the same i.e., sharing of information as fast as possible in terms of medium such as electrical cables, fiber optic cables and wireless radio networks. This paper deals regarding efficiency and compatibility of routing protocols based on QoS metrics in a wireless environment. A Mobile Adhoc Network (MANET) environment is created in Network Simulator-2 software, where the nodes move at a particular speed selecting the shortest path to reach the destination from the source. Performance of the network in terms of quality of services metrics are analyzed for different topologies ranging from 10 to 30 nodes with routing protocol such as AODV, DSDV and AOMDV. Main aim of this paper is to choose a particular routing protocol for the topology considered under different scenarios without compromising on the QoS parameters. The graphical analysis is made in the paper based on the routing parameters such as network size, throughput, Packet Delivery Ratio (PDR), End to End delay, and Jitter.

Keywords: AODV, AOMDV, DSDV, Jitter, MANET, PDR, QoS, Throughput

Date of Submission: 16-04-2018 Date of acceptance: 11-05-2018

I. INTRODUCTION

Mobile ad hoc network (MANET) which is infrastructure less, consists of mobile nodes and wireless machine nodes connected to each other. There is no centralized administration mechanism in MANETs. In MANET every node acts as a router. In MANET, a node can send packets directly to another node if both the nodes are in their respective transmission ranges or else packet transmission can be multihop. Due to arbitrary movement of nodes, network topology changes rapidly. Since the nodes are changing much rapidly it is very difficult for the routing protocols to perform their task correctly. So we use topology approximation protocols to overcome the difficulty.

The limited resources in MANETs have made designing of an efficient and reliable routing strategy a very challenging problem. An intelligent routing strategy is required to efficiently use the limited resources while at the same time being adaptable to the changing network conditions such as: network size, traffic density, network partitioning. In parallel with this, the routing protocol may need to provide different levels of QoS to different types of applications and users. Prior to the increased interest in wireless networking, in wired networks two main algorithms called link state and distance vector algorithms were used [5-6].

In link state routing each node maintains an updated view of the network by periodically broadcasting the link state costs of its neighbouring nodes to all other nodes using a flooding strategy. When each node receives an update packet, they update their view of the network and their link state information by applying a shortest path algorithm to choose the next hop node for each destination.

In distance vector routing algorithm, for each destination *x*, each node *i* maintains a set of distances $D^x ij$ where j ranges over the neighbours of node i. Node 'i' selects a neighbor k, to be the next hop for x if $D^x ij = \min j \{D^x ij\}$ This allows each node to select the shortest path to each destination. The distance vector information is updated at each node by a periodical dissemination of the current estimate of the shortest distance to every node.

The traditional link state and the distance vector do not scale in large MANETs. This is mainly due to periodic or frequent route updates in large networks may consume significant part of the bandwidth, increase channel orientation and may require each node to frequently recharge their power supply.

To overcome the problem associated with link state and distance vector algorithms a number of routing protocols have been proposed for MANETs. These protocols can be classified into three different groups: Global/proactive, on-demand/ reactive and hybrid. In proactive routing protocols, the routes to all the destinations (or parts of the network) are determined at the start up and maintained by using a periodic route

update process. In reactive routing protocols, routes are determined when they are required by the source using a route discovery process. Hybrid routing protocols combine the basic properties of the first two classes of protocols into one.

II. LITERATURE SURVEY

Quality of service (QoS) support in mobile ad hoc networks (MANETs) is a challenging task, due to resource constraints and dynamic topology of it. Although lots of research have been done on supporting QoS in the Internet and other networks, they are not suitable for the MANET environment. While traditional protocol layering is an important abstraction to reduce complexity of network design in wired networks, it is not well suited to wireless networks to provide complex functionalities like QoS due to interdependencies of different layers. A thorough investigation of the different techniques at different layers and their adaptation has lead to this simple framework which gives better performance under different traffic loads and mobility scenarios. An ad hoc network is a rapidly deployable wireless network that has no centralized control mechanism. Mobile devices in an ad hoc network request a variety of data types, including text and multimedia data. Different types of data need to be treated with different qualities of service. The implementation of service differentiation in wireless networks is very difficult because of device mobility and wireless channel contention.

Supporting high volume of data transmission in a highly dynamic architecture like Mobile Ad hoc Networks (MANET) still remains a major point of research. We have organized this work in two logical steps. In the first part, we have described an agent-based framework with its associated protocols and mechanisms. The primary objective of this mobile multi-agent framework is to make all nodes in the system topology-aware. The second part of the work attempts the agent-enabled proactive replenishment of fresh topology information enables each node to constantly evaluate network conditions and to take decisions on adaptive route selection [5-6].

AODV routing protocol: IEFT community has published the first version of the AODV Routing Protocol (Ad hoc On Demand Distance Vector). AODV belongs to the class of Distance Vector Routing Protocols (DV). In a DV every node knows its neighbours and the costs to reach them. A node maintains its own routing table, storing all nodes in the network, the distance and the next hop to them. If a node is not reachable the distance to it is set to infinity. Every node sends its neighbours periodically its whole routing table. So they can check if there is a useful route to another node using this neighbour as next hop. When a link breaks a Count-To-Infinity could happen. AODV is an 'on demand routing protocol' with small delay. That means that routes are only established when needed to reduce traffic overhead. AODV supports Unicast, Broadcast and Multicast without any further protocols. In AODV the routing table is expanded by a sequence number to every destination and by time to live for every entry.

AODV has the following characteristics such as On demand (with small delay), Unicast / Multicast / Broadcast provided, Loop free, Quick aging, Link breakages efficiently repaired, Distributed Routing, hop-by-hop, Deterministic and Single path

DSDV routing protocol: Destination-Sequenced Distance-Vector Routing (DSDV) is a table-driven routing scheme for ad hoc mobile networks based on the Bellman–Ford algorithm. It was developed by C. Perkins and P.Bhagwat [10] in 1994. The main contribution of the algorithm was to solve the routing loop problem. Each entry in the routing table contains a sequence number, the sequence numbers are generally even if a link is present; else, an odd number is used. The number is generated by the destination, and the emitter needs to send out the next update with this number. Routing information is distributed between nodes by sending full *dumps* infrequently and smaller incremental updates more frequently. If a router receives new information, then it uses the latest sequence number. If the sequence number is the same as the one already in the table, the route with the better metric is used. Stale entries are those entries that have not been updated for a while. Such entries as well as the routes using those nodes as next hops are deleted. DSDV requires a regular update of its routing tables, which uses up battery power and a small amount of bandwidth even when the network is idle. Whenever the topology of the network changes, a new sequence number is necessary before the network re-converges; thus, DSDV is not suitable for highly dynamic networks.

AOMDV routing protocol: AOMDV stands for Ad-hoc On-demand Multipath Distance Vector Routing protocol. AOMDV is a multipath extension to the AODV protocol. In AOMDV protocols multiple routes are founded between the source and destination. It uses alternate routes on a route failure. In AOMDV protocols new route discovery is needed when all the routes fail. In AOMDV protocols multipath routing is the enhancement of unipath routing in which advantage is to handle the load in network and avoid the possibility of congestion and increases reliability.

Advantages of AOMDV protocol are it establishes route on demand, it creates loop free nodes, it maintains connectivity and fast and efficient recovery from failures. Disadvantages of AOMDV protocol is that it has more message overheads during route discovery due to increased flooding and since it is a multipath routing protocol, the destination replies to the multiple RREQs those results in a longer overhead packets in response to single RREQ packet may lead to heavy control overhead.

III. SOFTWARE DEVELOPMENT

Software Structure and Mechanism in NS-2: The key to get to know ns-2 [1] is it is a discrete event network simulator. In ns-2 network physical activities are translated to events, events are queued and processed in the order of their scheduled occurrences. And the simulation time progresses with the events processed. And also the simulation "time" may not be the real life time as we "inputted". Typically, it can configure transport layer protocols, routing protocols, interface queues, and also link layer mechanisms. We can easily see that this software tool in fact could provide us a whole view of the network construction, meanwhile, it also maintain the flexibility for us to decide. Thus, just this one software can help us simulate nearly all parts of the network. This definitely will save us great amount of cost invested on network constructing. The following Figure 1 shows a layered structure which ns-2 can simulate [2].

After the simulation finishes, ns-2 presents the detailed information to the network layer by means of a trace file which is a line by line account of all the events. This shows the significance of an event driven mechanism which records the events as they occur. These records can then be traced to evaluate the performance of critical things in the network such as routing protocol, MAC layer load, etc

.

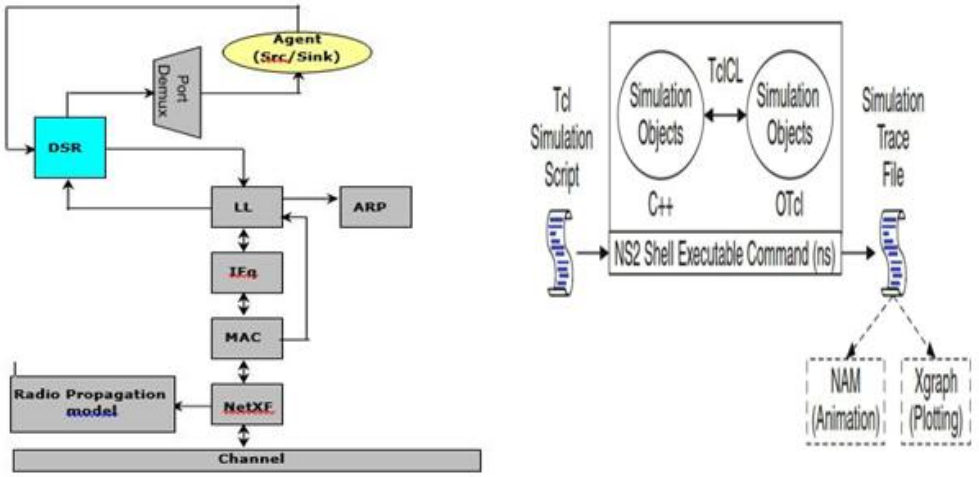

Figure 1(a) & 1(b): NS-2 simulate layered structure of network Figure and Data flow for one time simulation

Figure 1(a) & 1(b): NS-2 simulate layered structure of network Figure and Data flow for one time simulation. Figure 1(b) shows, the data flow of one time simulation in ns-2, the user inputs an OTcl source file, the OTcl script does the work of initiating an event scheduler, sets up the network topology using the network objects and the plumbing functions in the library, and tells traffic sources when to start and stop transmitting packets through the event scheduler. And then, this OTcl script file is passed to ns-2, in this view, we can treat ns-2 as Object-oriented Tcl (OTcl) script interpreter that has a simulation event scheduler and network component object libraries, and network setup module libraries. And then the detail network construction and traffic simulation will be actually done in ns-2. After a simulation is finished, NS produces one or more text-based output files that contain detailed simulation data, and the data can be used for simulation analysis.

The above figure shows the ns-2 developer's view, the layered structure of ns. The event schedulers and the network components are implemented in C++ and is made available to the TCL script, thus the lowest level of ns-2 is implemented in C++, and the TCL script is placed at the top to make simulation easier. We see the overview of the network on top of the TCL level. This defines the simulation scenario. All these things combined is the ns-2 software.

To successfully carry out one simulation, we must first tell ns-2 things it may need from us for one simulation. So what we need is the follow three necessary items:

1) Appearance of the network: the whole topology view of sensor network or mobile network, this includes the position of nodes with (x, y, z) coordinate, the node movement parameters, the movement starting time,

the movement is to what direction, and the node movement speed with pausing time between two supposed movement.

2) Internal of the network: Since the simulation is on the network traffic, so it is important we tell the ns2 about which nodes are the sources, how about the connections, what kind of

connection we want to use.

3) Configuration of the layered structure of each node in the network, this includes the detail configuration of network components on sensor node, and also we need to drive the simulation, so we need to give out where to give out the simulation results which is the trace file, and how to organize a simulation process.

TCL script to run a simple wireless simulation

In this section, we then will present step by step process of writing the TCL code [3-9] for simulation in ns-2:

Step 1. Create an instance of the simulator:*set ns_ [new Simulator]*

Step.2. Setup trace support by opening file "trace_bbtr.tr" and call the procedure trace-all*Set tracefd [open trace_bbtr.tr w]$ns_ trace-all $tracefd*

Step 3. Create a topology object that keeps track of all the nodes within boundary *set topo [new Topography]*

Step 4. The topography is broken up into grids and the default value of grid resolution is 1. A different value can be passed as a third parameter to load_flatgrid {}. *$topoload_flatgrid $val(x) $val(y)*

Step 5. Create the object God(General Operations Director) that is used to store global information about the state of the environment, network or nodes. God object is called internally by MAC objects in nodes, so we must create god in every cases.*set god_ [create-god $val(nn)]*

Step 6. Node configuration API may consist of defining the type of addressing (flat/hierarchical etc.), for example, the type of adhoc routing protocol, Link Layer, MAC layer, Ifq etc.

```
$ns_ node-config -adhocRouting $val(rp) \
-llType $val(ll) \
-macType $val(mac) \
-ifqType $val(ifq) \
-ifqLen $val(ifqlen) \
-antType $val(ant) \
-propType $val(prop) \
-phyType $val (netif) \
-channel [new $val(chan)]\
-topoInstance $topo \
-agentTrace ON \
-routerTrace ON \
-macTrace OFF \
-movementTraceOFF
```

Step 7. Create nodes and the random-motion for nodes is disabled here, as we are going to provide node position and movement (speed & direction) directives next

```
for {set i 0} {$i< $val (nn) } {incr i} {
set node_($i) [$ns_ node]
$node_($i) random-motion 0# Disable random motion}
```

Step 8. Give nodes positions to start with, Provide initial (X,Y, for now Z=0) co-ordinates for node_(0) and node_(1) and the rest of the nodes. Node0 has a starting position of (5,2) while Node1 starts off at location (390,385).

```
$node_(0) set X_ 5.0
$node_(0) set Y_ 2.0
$node_(0) set Z_ 0.0
$node_(1) set X_ 390.0
$node_(1) set Y_ 385.0
$node_(1) set Z_ 0.0
```

Similarly the positions are assigned to the rest of the nodes.

Step 9. Setup node movement. Say, at time 50.0s, node 1 starts to move towards the destination (x=25, y=20) at a speed of 15m/s. This API is used to change direction and speed of movement of nodes.*$ns_ at 50.0 "$node_ (1) setdest 25.0 20.0 15.0"*
Step 10. Setup traffic flow between the two nodes as follows: TCP connections between node_(0) and node_(1)

```
set tcp [new Agent/TCP]
$tcp set set sink [new Agent/TCPSink]
$ns_ attach-agent $node_(0) $tcp
$ns_ attach-agent $node_(1) $sink
$ns_ connect
$tcp $sink
set ftp [new Application/FTP] $ftp attach-agent
$tcp $ns_ at 10.0 "$ftp start"
```

Step 11. Define stop time when the simulation ends and tell nodes to reset which actually resets their internal network components. In the following case, at time 150.0s, the simulation shall stop. The procedure stop{} is called to flush out traces and close the trace file.

```
for {set i 0} {$i< $val(nn) } {incr i} {$ns_ at 150.0 "$node_($i) reset";  }
$ns_ at 150.0001 "stop"
$ns_ at 150.0002 "puts \"NS EXITING...\" ; $ns_ halt"
proc stop {} {
global ns_tracefd
nf $ns_ flushtrace
close $tracefd close $nf}
```

Step 12. Finally the command to start the simulation *puts "Starting Simulation...\n" $ns_ run*

So, these 12 steps could finish one time simulation. However, there exist some problems on such kind of use on typical network performance test situations. Performance testing usually needs to be scalable in the number of nodes and network transmitting packets. Suppose for one network there are hundreds of nodes, we need to set all of the nodes' positions and their movement, this a huge amount of workload, also, suppose we need to setup all the possible sources and destinations and even connections, also is a huge workload, Furthermore, even if we can set them, we cannot guarantee our input is randomly selected, which is necessary for a fair comparison.

IV. SIMULATION RESULTS

Simulation results are shown for 10 nodes and 30 nodes to discuss the behavior of the routing protocols in small and large network scenarios. The parameters that are considered that makes AODV, AOMDV routing protocols different from its parent protocol i.e. DSDV are Throughput vs average RTT, Cumulative sum of dropped packets, Cumulative sum of the generated packets Jitter

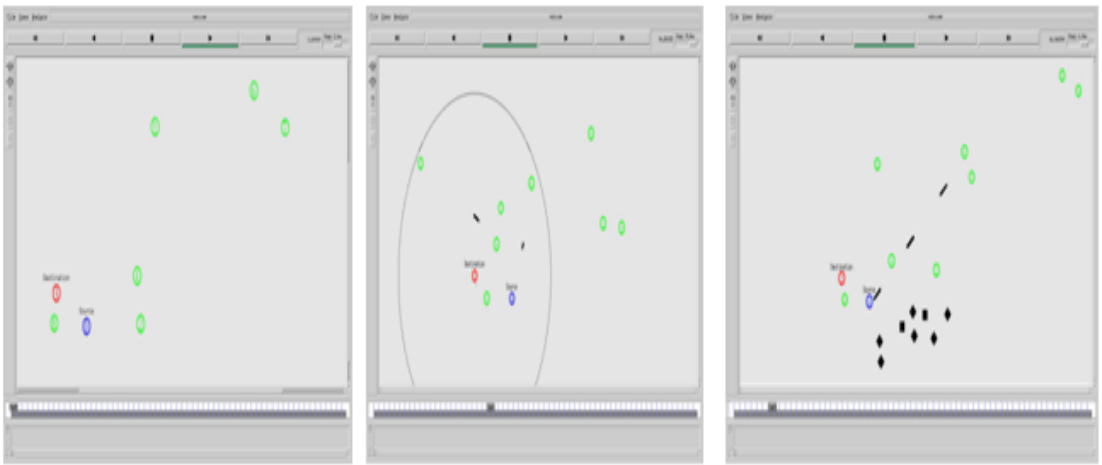

Figure 2: Network with 10 nodes running on AODV, AOMDV, DSDV routing protocols

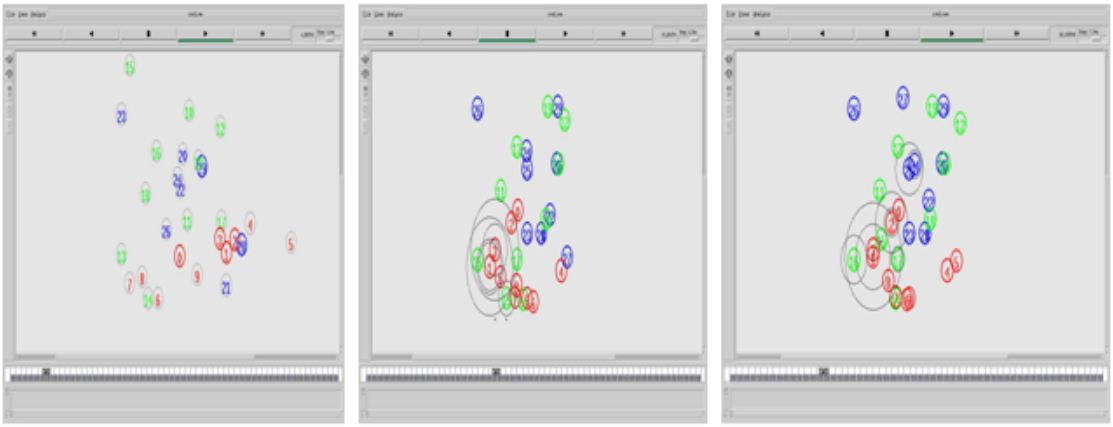

Figure 3 Network running with 30 nodes on AODV, AOMDV, DSDV routing protocols

Throughput Analysis: Throughput or network throughput is the rate of *successful* message delivery over a communication channel. This data may be delivered over a physical or logical link, or pass through a certain network node. The throughput is usually measured in bits per second (bit/s or bps), and sometimes in data packets per second or data packets per time slot.

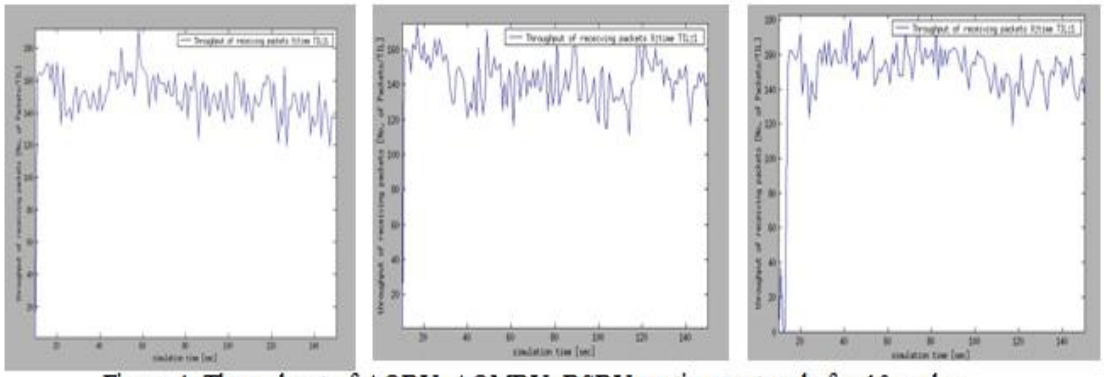

Figure 4: Throughput of AODV, AOMDV, DSDV routing protocols for 10 nodes

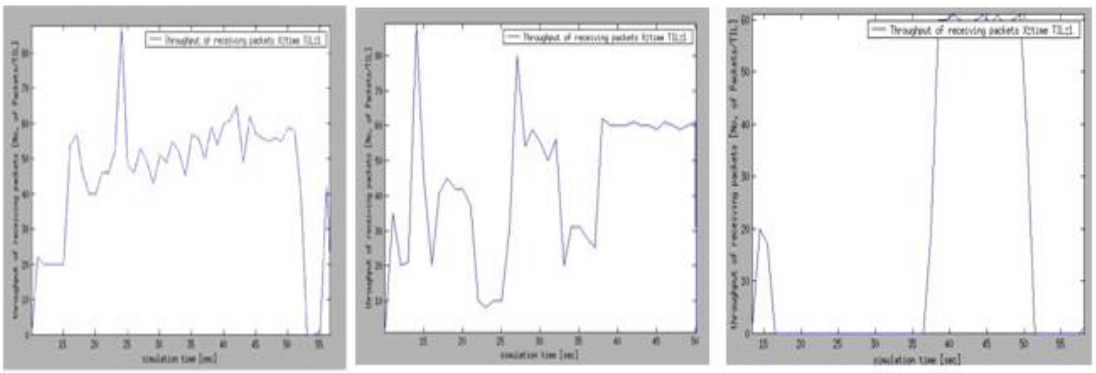

Figure 5: Throughput of AODV, AOMDV, DSDV routing protocols for 30 nodes

On observing the graph figures, we can see that the throughput of the AODV network for 10 nodes is constantly high during the simulation time whereas it is lesser initially and pickups up in the case of DSDV, and in the case of 30 nodes AOMDV is more compared to AODV and DSDV, this is because of the delay caused in searching for the optimum route to the destination by verifying the dynamic vector tables.

Packet Analysis

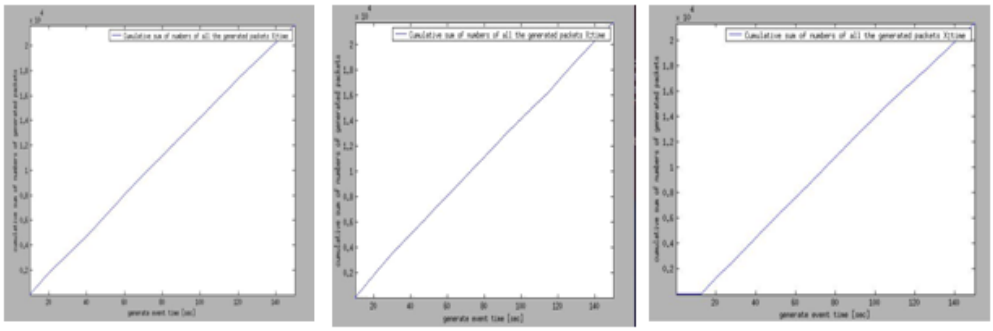

Figure 6: Cumulative sum of the generated packets (AODV, AOMDV, DSDV routing protocols) for 10 nodes

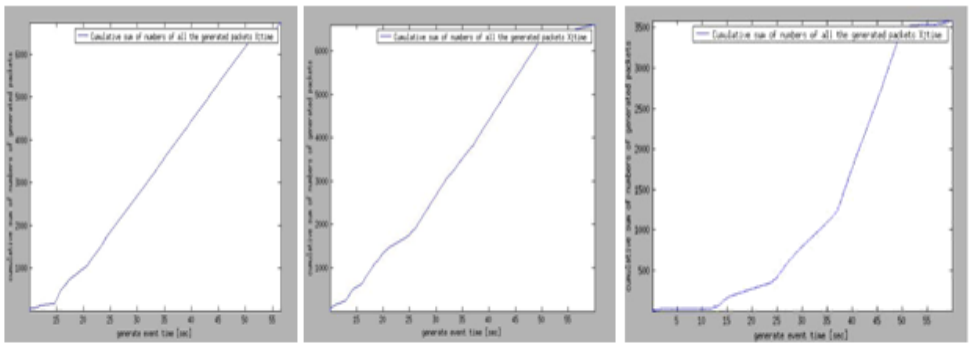

Figure 7: Cumulative sum of the generated packets (AODV, AOMDV, DSDV routing protocols) for 30 nodes

The above figures 5.3.2.1, 5.3.2.2, 5.3.2.3, 5.3.2.4, 5.3.2.5, 5.3.2.6 form a part of the packet analysis for the network running on AODV, DSDV and AOMDV routing protocols respectively. The graphs compare the number of dropped packets in the entire simulation time where it is observed that the number of packets dropped for 10 nodes is less in both AODV and AOMDV routing protocols compared to no packets dropped in DSDV routing protocol. Where it is observed that the number of packets dropped for 30 nodes is less in both AOMDV routing protocol compared to no packets dropped in AODV & DSDV routing protocols.
This shows that AOMDV routing protocol is efficient than AODV & DSDV routing protocols for more number of nodes. The packets generated in AOMDV & AODV increase at the outset of communication in AODMV & AODV whereas the generated packets only increase after a delay in the case of DSDV.

Cumulative Distribution of Jitter

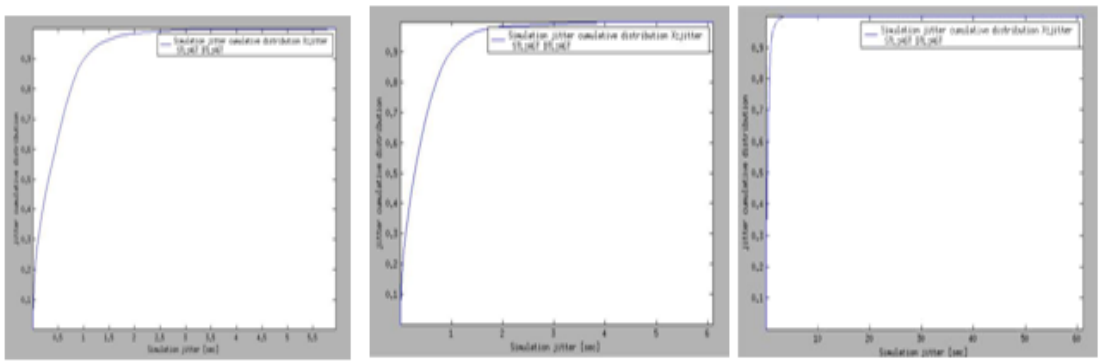

Figure 8: Jitter Cumulative distribution (AODV, AOMDV, DSDV routing protocols) for 10 nodes

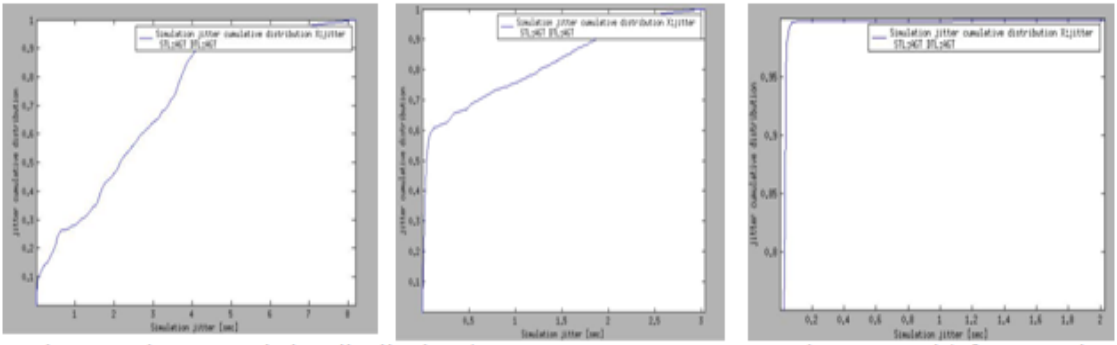

Figure 9: Jitter Cumulative distribution (AODV, AOMDV, DSDV routing protocols) for 30 nodes

Jitter is defined as a variation in the delay of received packets. At the sending side, packets are sent in a continuous stream with the packets spaced evenly apart. Due to network congestion, improper queuing, or configuration errors, this steady stream can become lumpy, or the delay between each packet can vary instead of remaining constant. The figures 8 & 9 depict the cumulative distribution of jitter over the simulation jitter (sec) for AODV, AOMDV and DSDV protocols respectively. Here we can observe that in all the cases jitter increases in an exponential manner but the overall jitter stabilizes faster in DSDV when compared to AODV & AOMDV.

Network Simulation Information

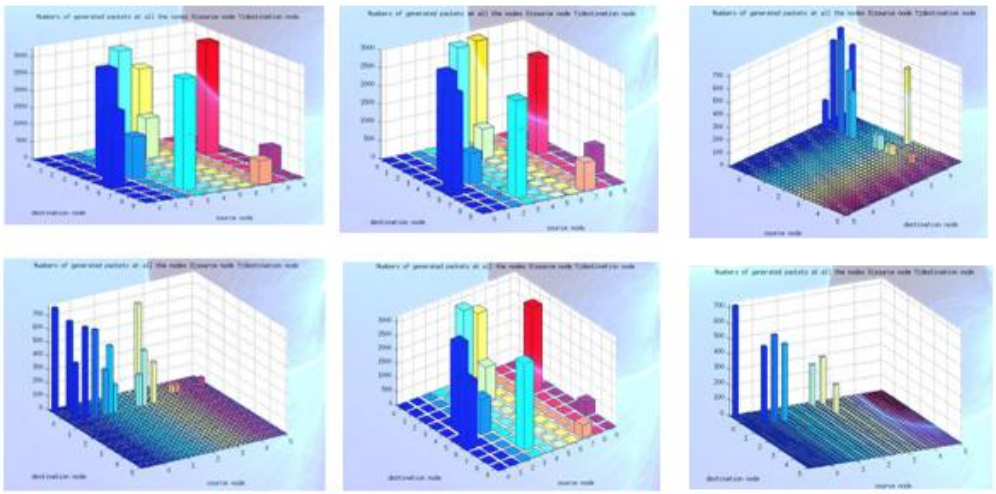

Figure 10: No. of generated packets for AODV, AOMDV, DSDV Routing Protocols with 10 & 30 nodes

V. CONCLUSION

Performance analysis among DSDV, AODV and AOMDV routing protocol depicts that the applications where throughput, residual energy are important and delay can be tolerated; then the AOMDV routing protocol can be the best solution. We also observed that in a high speed movement of nodes, AOMDV can be the best choice. Though AOMDV routing protocol performs better in our simulation environment considering energy consumption and throughput, it has some limitations like more delay, more routing load in the network.

AOMDV & AODV are -- efficient algorithms for ad-hoc networks, highly scalable, need for broadcast is minimized, quick response to link breakage in active routes and loop free routes. Some of the conclusions from the analysis are:

Jitter of dropped packets: AOMDV
Jitter of forwarded packets: AODV
Throughput of generating packets: AODV
Throughput of sending packets: AOMDV
Throughput of receiving packets: DSDV
Throughput of sending packets vs End to End simulation: AOMDV

Out of all routing protocols AOMDV generates highest number of packets than AODV and DSDV.
Advantages: The main advantage of AOMDV and AODV protocol are having routes established on demand and that destination sequence numbers are applied for find the latest route to the destination. The connection setup delay is lower.

Disadvantages: One disadvantage of AODV protocol is that intermediate nodes can lead to inconsistent routes if the source sequence number is very old and the intermediate nodes have a higher but not the latest destination sequence number, thereby having stale entries. Also, multiple Route Reply packets in response to a single Route Request packet can lead to heavy control overhead. Another disadvantage of AODV is unnecessary bandwidth consumption due to periodic beaconing.

DSDV gives comparatively lower throughput as the large number of routing bits is required. Increase in overhead reduces the throughput. The difference in the routing load of AODV and DSDV decreases with an increase in the load.

BIBLIOGRAPHY

[1]. The network simulator - ns-2. http://www. isi.edu/nsnam/ns/.
[2]. K. Fall and K. Varadhan (Eds.), ns notes and documentation, http://wwwmash.cs.berkeley.edu/ns/.
[3]. NS by Example, http://nile.wpi.edu/NS/
[4]. NS-2 Trace Formats. http://k-lug.org/~griswold/NS2/ns2-trace-formats.html
[5]. Computer Networks, Fourth Edition, Andrew S. Tanenbaum
[6]. Mehran Abolhasan a, Tadeusz Wysocki a, Eryk Dutkiewicz,‟ A review of routing protocols for mobile ad hoc networks", Ad Hoc Networks 2 (2004) 1–22.
[7]. M. S. Corson and A. Ephreruides, “A distributed routing algorithm for mobile wireless networks", ACM L Wireless Network, vol. 1, no. 1, pp. 61+1, 1995.
[8]. S. Corson and J. Macker, "Mobile ad hoc networking Routing protocol performance issues and evaluation considerations (internet-Draft)V", Mar. 1998.
[9]. Bency Wilson, Geethu Bastian, Vinitha Ann Regi, Arun Soman, "EZR: Enhanced Zone Based Routing in Manet", International Journal of Modern Engineering Research, Vol. 3, Issue. 3, May.-June. 2013 pp-1832-1836.
[10]. C. Perkins and P.Bhagwat, "Highly Dynamic Destination Sequenced Distance Vector Routing DSDV for Mobile Computers", Proceedings of SIGCOMM 1994, pp 234-244

IOSR Journal of Engineering (IOSRJEN) is UGC approved Journal with Sl. No. 3240, Journal no. 48995.

N Dinesh Kumar, VSK Reddy, "Performance Analysis of Routing Protocols", IOSR Journal of Engineering (IOSRJEN), vol. 08, no. 5, 2018, pp. 69-77.