**CSE1004**

**NETWORKS AND COMMUNICATION**

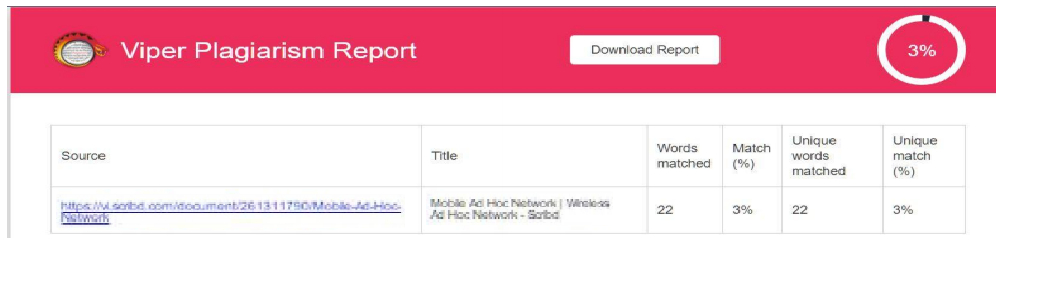
**REPORT (REVIEW-3)**

**TOPIC:   
Simulate Mobile Adhoc Network (MANET) using ns2**

Under the guidance of

**Dr. Ganesh Gopal D**

**M.S.SANJAY-15BCE0517**

**ABSTRACT:**

* Networks that support the ad hoc architecture are typically called wireless ad hoc networks or mobile ad hoc networks (MANET) these two terms are used interchangeably.
* Such networks are typically assumed to be self-forming and self-healing. This project aim is to simulate the wireless ad hoc network using NS2 simulator.
* There is no static infrastructure such as base station. If two hosts are not within radio range, all message communication between them must pass through one or more intermediate hosts that double as routers.

**INTRODUCTION:**

In the recent years, wireless technology has got popularity and usage, thus opening new fields of applications in the domain of networking. The wireless networks may categorize as infrastructure based and infrastructure less network. In infrastructure based networks the mobile nodes are communicated with the help of common access point. Where as in infrastructure less network the mobile nodes are communicated cooperatively with each other, where the participating nodes do not rely on any existing network infrastructure such networks are known as wireless ad hoc networks. In wireless ad hoc networks each node is functions as node as well as router.

Ad hoc networks have a wide array of military and commercial applications. They are ideal in situations where installing an infrastructure network is not possible or when the purpose of the network is too transient or even for the reason that the previous infrastructure network was destroyed.

**AIM:**

We simulate the network which uses AODV as routing protocol and finds the network parameter values and the performance of the AODV protocol is measured.

**OBJECTIVE:**

The objective of the project is to implement Adhoc on demand Distance Vector Routing-aodv protocol in a Mobile Adhoc NETWORK-MANET using network stimulator 2.

**PROJECT SCOPE:**

* Ad hoc networks are very well suited for many situations, in which an infrastructure network can’t be built or it is impossible to build an infrastructure.
* The interest of ad hoc networks increases rapidly in recent year, because ad hoc supports mobility and freedom in the networks.
* Data can be exchanged without cable, access point, or portable memory space.
* Now a day’s computers and phones manufacturers implement ad hoc technology to their products.
* Tactical Networks: -Military communication automated system.
* Entertainment:- Multi users games, Robotics pets.
* Emergency Services: - Disaster recovery, Earth quakes.
* Sensor Network: -Earth activities, Remote weather for sensors.

**HARDWARE/SOFTWARE DESCRIPTION:**

Software used is network stimulator 2.It runs on Linux operating system.

No other special hardware is required.

**PROJECT DESCRIPTION NOTE:**

The project uses ns2 for implementation. Network Simulator version 2 widely known as NS-2 is a discrete event driven network simulation tool for simulation of network to study the dynamic nature of network. Ad-hoc On Demand Distance Vector: AODV is a distance vector type routing. It does not require nodes to maintain routes to destinations that are not actively used. As long as the endpoints of a communication connection have valid routes to each other, AODV does not play a role.

**WHAT IS MANET?**

* A mobile adhoc network (MANET) is a continuously self-configuring, infrastructure-less network of mobile devices connected wirelessly.
* Each device in a MANET is free to move independently in any direction, and will therefore change its links to other devices frequently. Each must forward traffic unrelated to its own use, and therefore be a router. The primary challenge in building a MANET is equipping each device to continuously maintain the information required to properly route traffic.

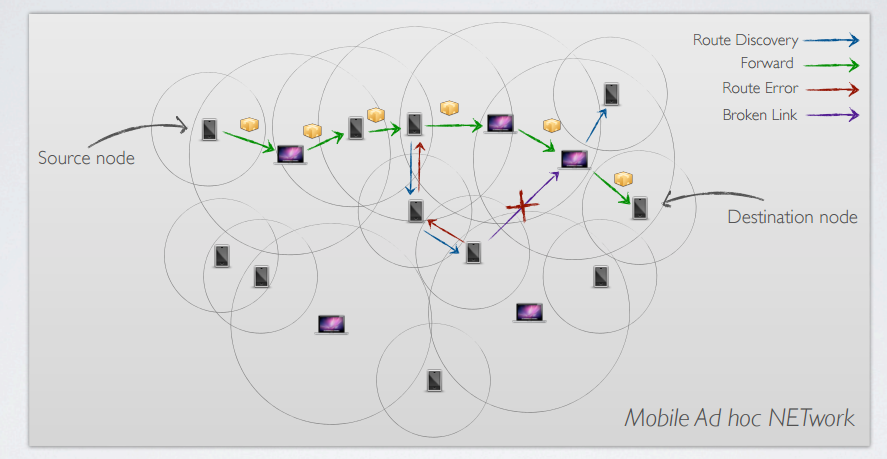
**Example scenarios for MANET s:**

* Meetings „
* Emergency or disaster relief situations „ Military communications „
* Wearable computers „
* Sensor networks

**TYPES OF AD-HOC NETWORKS:**

* Vehicular ad hoc networks (VANETs) are used for communication between vehicles and roadside equipment. Intelligent vehicular ad hoc networks (InVANETs) are a kind of artificial intelligence that helps vehicles to behave in intelligent manners during vehicle-to-vehicle collisions, accidents.
* Smart phone ad hoc networks (SPANs) leverage the existing hardware (primarily Bluetooth and Wi-Fi) in commercially available smart phones to create peer-to-peer networks without relying on cellular carrier networks, wireless access points, or traditional network infrastructure.
* Internet-based mobile ad-hoc networks (iMANETs) are ad hoc networks that link mobile nodes and fixed Internet-gateway nodes. For example, multiple sub-MANETs may be connected in a classic Hub-Spoke VPN to create a geographically distributed MANET.

**ARCHITECTURE DIAGRAM OF ADHOC NETWORK:**



**AD HOC ROUTING PROTOCOLS:**

**Targets:**

Minimize number of exchanged messages

Simplify processing and routing operations

Keeping updated the network scenario

Prevent loops inside the network

**METHODS:**

AODV defines three types of control messages for route maintenance:

**RREQ** - A route request message is transmitted by a node requiring a route to a node.

As an optimization AODV uses an expanding ring technique when flooding these messages. Every RREQ carries a time to live (TTL) value that states for how many hops this message should be forwarded. This value is set to a predefined value at the first transmission and increased at retransmissions. Retransmissions occur if no replies are received.

Data packets waiting to be transmitted(i.e. the packets that initiated the RREQ) should be buffered locally and transmitted by a FIFO principal when a route is set.

**RREP** - A route reply message is unicasted back to the originator of a RREQ if the receiver is either the node using the requested address, or it has a valid route to the requested address. The reason one can unicast the message back, is that every route forwarding a RREQ caches a route back to the originator.

**RERR** - Nodes monitor the link status of next hops in active routes. When a link breakage in an active route is detected, a RERR message is used to notify other nodes of the loss of the link. In order to enable this reporting mechanism, each node keeps a ``precursor list'', containing the IP address for each its neighbors that are likely to use it as a next hop towards each destination

**ALGORITHM:**

Route Discovery:

When a source node needs a route to a destination node and there is not the valid route in the routing table, the source node broadcasts a route request packet (RREQ) to the destination node. When each node receives the RREQ, it creates or updates a reverse route to the source node in the routing table. If it does not have a valid route to the destination node in the routing table, it rebroadcasts the RREQ. When the RREQ flooding from the source node arrives at the destination node, the destination node creates or updates the reverse route. And it unicasts a route reply packet (RREP) which has an incremented the sequence number to the reverse route. When each node receives the RREP, it creates or updates a forward route to the destination node and it forwards the RREP to the reverse route. When the RREP arrives at the source node along with the reverse route, it creates or updates the forward route, and starts communications.

Route Maintenance :

Each node broadcasts a Hello packet periodically for local connectivity. It broadcasts the RREP with TTL=1 as the Hello packet. When the node does not receive any packets from a neighbor during a few seconds, it assumes a link break to the neighbor. In addition, when the node has the link break to the neighbor based on an acknowledgment of MAC layer, it detects a route break to the destination node that the next hop of the route is the neighbor. When the node that detects the link break is close to the destination node (that is to say the number of hops to the destination node is smaller than the number of hops to the source node), it requires a new route to the destination node, which is known as Local Repair.

**CODING:**

set opt(chan) Channel/WirelessChannel ;# channel type

set opt(prop) Propagation/TwoRayGround ;# radio-propagation model

set opt(netif) Phy/WirelessPhy ;# network interface type

set opt(mac) Mac/802\_11 ;# MAC type

set opt(ifq) Queue/DropTail/PriQueue ;# interface queue type

set opt(ll) LL ;# link layer type

set opt(ant) Antenna/OmniAntenna ;# antenna model

set opt(ifqlen) 50 ;# max packet in ifq

set opt(nn) 22 ;# number of mobilenodes

set opt(rp) AODV ;# routing protocol

set opt(x) 1800 ;# X dimension of topography

set opt(y) 840 ;# Y dimension of topography

### Setting The Simulator Objects

set ns\_ [new Simulator]

#create the nam and trace file:

set tracefd [open aodv.tr w]

$ns\_ trace-all $tracefd

set namtrace [open aodv.nam w]

$ns\_ namtrace-all-wireless $namtrace $opt(x) $opt(y)

set topo [new Topography]

$topo load\_flatgrid $opt(x) $opt(y)

create-god $opt(nn)

set chan\_1\_ [new $opt(chan)]

#### Setting The Distance Variables

# For model 'TwoRayGround'

set dist(5m) 7.69113e-06

set dist(9m) 2.37381e-06

set dist(10m) 1.92278e-06

set dist(11m) 1.58908e-06

set dist(12m) 1.33527e-06

set dist(13m) 1.13774e-06

set dist(14m) 9.81011e-07

set dist(15m) 8.54570e-07

set dist(16m) 7.51087e-07

set dist(20m) 4.80696e-07

set dist(25m) 3.07645e-07

set dist(30m) 2.13643e-07

set dist(35m) 1.56962e-07

set dist(40m) 1.56962e-10

set dist(45m) 1.56962e-11

set dist(50m) 1.20174e-13

Phy/WirelessPhy set CSThresh\_ $dist(50m)

Phy/WirelessPhy set RXThresh\_ $dist(50m)

# Defining Node Configuration

$ns\_ node-config -adhocRouting $opt(rp) \

-llType $opt(ll) \

-macType $opt(mac) \

-ifqType $opt(ifq) \

-ifqLen $opt(ifqlen) \

-antType $opt(ant) \

-propType $opt(prop) \

-phyType $opt(netif) \

-topoInstance $topo \

-agentTrace ON \

-routerTrace ON \

-macTrace ON \

-movementTrace ON \

-channel $chan\_1\_

### Creating The WIRELESS NODES

set Server1 [$ns\_ node]

set Server2 [$ns\_ node]

set n2 [$ns\_ node]

set n3 [$ns\_ node]

set n4 [$ns\_ node]

set n5 [$ns\_ node]

set n6 [$ns\_ node]

set n7 [$ns\_ node]

set n8 [$ns\_ node]

set n9 [$ns\_ node]

set n10 [$ns\_ node]

set n11 [$ns\_ node]

set n12 [$ns\_ node]

set n13 [$ns\_ node]

set n14 [$ns\_ node]

set n15 [$ns\_ node]

set n16 [$ns\_ node]

set n17 [$ns\_ node]

set n18 [$ns\_ node]

set n19 [$ns\_ node]

set n20 [$ns\_ node]

set n21 [$ns\_ node]

set n22 [$ns\_ node]

set opt(seed) 0.1

set a [ns-random $opt(seed)]

set i 0

while {$i < 5} {

incr i

}

### Setting The Initial Positions of Nodes

$Server1 set X\_ 513.0

$Server1 set Y\_ 517.0

$Server1 set Z\_ 0.0

$Server2 set X\_ 1445.0

$Server2 set Y\_ 474.0

$Server2 set Z\_ 0.0

$n2 set X\_ 36.0

$n2 set Y\_ 529.0

$n2 set Z\_ 0.0

$n3 set X\_ 143.0

$n3 set Y\_ 666.0

$n3 set Z\_ 0.0

$n4 set X\_ 201.0

$n4 set Y\_ 552.0

$n4 set Z\_ 0.0

$n5 set X\_ 147.0

$n5 set Y\_ 403.0

$n5 set Z\_ 0.0

$n6 set X\_ 230.0

$n6 set Y\_ 291.0

$n6 set Z\_ 0.0

$n7 set X\_ 295.0

$n7 set Y\_ 419.0

$n7 set Z\_ 0.0

$n8 set X\_ 363.0

$n8 set Y\_ 335.0

$n8 set Z\_ 0.0

$n9 set X\_ 334.0

$n9 set Y\_ 647.0

$n9 set Z\_ 0.0

$n10 set X\_ 304.0

$n10 set Y\_ 777.0

$n10 set Z\_ 0.0

$n11 set X\_ 412.0

$n11 set Y\_ 194.0

$n11 set Z\_ 0.0

$n12 set X\_ 519.0

$n12 set Y\_ 361.0

$n12 set Z\_ 0.0

$n13 set X\_ 569.0

$n13 set Y\_ 167.0

$n13 set Z\_ 0.0

$n14 set X\_ 349.0

$n14 set Y\_ 546.0

$n14 set Z\_ 0.0

$n15 set X\_ 466.0

$n15 set Y\_ 668.0

$n15 set Z\_ 0.0

$n16 set X\_ 489.0

$n16 set Y\_ 794.0

$n16 set Z\_ 0.0

$n17 set X\_ 606.0

$n17 set Y\_ 711.0

$n17 set Z\_ 0.0

$n18 set X\_ 630.0

$n18 set Y\_ 626.0

$n18 set Z\_ 0.0

$n19 set X\_ 666.0

$n19 set Y\_ 347.0

$n19 set Z\_ 0.0

$n20 set X\_ 741.0

$n20 set Y\_ 152.0

$n20 set Z\_ 0.0

$n21 set X\_ 882.0

$n21 set Y\_ 264.0

$n21 set Z\_ 0.0

$n22 set X\_ 761.0

$n22 set Y\_ 441.0

$n22 set Z\_ 0.0

## Giving Mobility to Nodes

$ns\_ at 0.75 "$n2 setdest 379.0 349.0 20.0"

$ns\_ at 0.75 "$n3 setdest 556.0 302.0 20.0"

$ns\_ at 0.20 "$n4 setdest 309.0 211.0 20.0"

$ns\_ at 1.25 "$n5 setdest 179.0 333.0 20.0"

$ns\_ at 0.75 "$n6 setdest 139.0 63.0 20.0"

$ns\_ at 0.75 "$n7 setdest 320.0 27.0 20.0"

$ns\_ at 1.50 "$n8 setdest 505.0 124.0 20.0"

$ns\_ at 1.25 "$n9 setdest 274.0 487.0 20.0"

$ns\_ at 1.25 "$n10 setdest 494.0 475.0 20.0"

$ns\_ at 1.25 "$n11 setdest 899.0 757.0 25.0"

$ns\_ at 0.50 "$n12 setdest 598.0 728.0 25.0"

$ns\_ at 0.25 "$n13 setdest 551.0 624.0 25.0"

$ns\_ at 1.25 "$n14 setdest 397.0 647.0 25.0"

$ns\_ at 1.25 "$n15 setdest 748.0 688.0 25.0"

$ns\_ at 1.25 "$n16 setdest 842.0 623.0 25.0"

$ns\_ at 1.25 "$n17 setdest 678.0 548.0 25.0"

$ns\_ at 0.75 "$n18 setdest 741.0 809.0 20.0"

$ns\_ at 0.75 "$n19 setdest 437.0 799.0 20.0"

$ns\_ at 0.20 "$n20 setdest 159.0 722.0 20.0"

$ns\_ at 1.25 "$n21 setdest 700.0 350.0 20.0"

$ns\_ at 0.75 "$n22 setdest 839.0 444.0 20.0"

## Setting The Node Size

$ns\_ initial\_node\_pos $Server1 125

$ns\_ initial\_node\_pos $Server2 125

$ns\_ initial\_node\_pos $n2 70

$ns\_ initial\_node\_pos $n3 70

$ns\_ initial\_node\_pos $n4 40

$ns\_ initial\_node\_pos $n5 70

$ns\_ initial\_node\_pos $n6 70

$ns\_ initial\_node\_pos $n7 70

$ns\_ initial\_node\_pos $n8 70

$ns\_ initial\_node\_pos $n9 70

$ns\_ initial\_node\_pos $n10 70

$ns\_ initial\_node\_pos $n11 70

$ns\_ initial\_node\_pos $n12 70

$ns\_ initial\_node\_pos $n13 70

$ns\_ initial\_node\_pos $n14 70

$ns\_ initial\_node\_pos $n15 70

$ns\_ initial\_node\_pos $n16 70

$ns\_ initial\_node\_pos $n17 70

$ns\_ initial\_node\_pos $n18 70

$ns\_ initial\_node\_pos $n19 70

$ns\_ initial\_node\_pos $n20 70

$ns\_ initial\_node\_pos $n21 70

$ns\_ initial\_node\_pos $n22 70

#### Setting The Labels For Nodes

$ns\_ at 0.0 "$Server1 label Server1"

$ns\_ at 0.0 "$Server2 label Server2"

$ns\_ at 0.0 "$n2 label node2"

$ns\_ at 0.0 "$n3 label node3"

$ns\_ at 0.0 "$n4 label node4"

$ns\_ at 0.0 "$n5 label node5"

$ns\_ at 0.0 "$n6 label node6"

$ns\_ at 0.0 "$n7 label node7"

$ns\_ at 0.0 "$n8 label node8"

$ns\_ at 0.0 "$n9 label node9"

$ns\_ at 0.0 "$n10 label node10"

$ns\_ at 0.0 "$n11 label node11"

$ns\_ at 0.0 "$n12 label node12"

$ns\_ at 0.0 "$n13 label node13"

$ns\_ at 0.0 "$n14 label node14"

$ns\_ at 0.0 "$n15 label node15"

$ns\_ at 0.0 "$n16 label node16"

$ns\_ at 0.0 "$n17 label node17"

$ns\_ at 0.0 "$n18 label node18"

$ns\_ at 0.0 "$n19 label node19"

$ns\_ at 0.0 "$n20 label node20"

$ns\_ at 0.0 "$n20 label node21"

$ns\_ at 0.0 "$n22 label node22"

$n2 color green

$ns\_ at 0.0 "$n2 color green"

$n3 color green

$ns\_ at 0.0 "$n3 color green"

$n4 color green

$ns\_ at 0.0 "$n4 color green"

$n5 color green

$ns\_ at 0.0 "$n5 color green"

$n6 color green

$ns\_ at 0.0 "$n6 color green"

$n7 color green

$ns\_ at 0.0 "$n7 color green"

$n8 color green

$ns\_ at 0.0 "$n8 color green"

$n9 color yellow

$ns\_ at 0.0 "$n9 color yellow"

$n10 color yellow

$ns\_ at 0.0 "$n10 color yellow"

$n11 color yellow

$ns\_ at 0.0 "$n11 color yellow"

$n12 color pink

$ns\_ at 0.0 "$n12 color pink"

$n13 color pink

$ns\_ at 0.0 "$n13 color pink"

$n14 color pink

$ns\_ at 0.0 "$n14 color pink"

$n15 color pink

$ns\_ at 0.0 "$n15 color pink"

$n16 color pink

$ns\_ at 0.0 "$n16 color pink"

$n17 color orange

$ns\_ at 0.0 "$n17 color orange"

$n18 color orange

$ns\_ at 0.0 "$n18 color orange"

$n19 color orange

$ns\_ at 0.0 "$n19 color orange"

$n20 color orange

$ns\_ at 0.0 "$n20 color orange"

$n21 color orange

$ns\_ at 0.0 "$n21 color orange"

$n22 color orange

$ns\_ at 0.0 "$n22 color orange"

$Server1 color maroon

$ns\_ at 0.0 "$Server1 color maroon"

$Server2 color maroon

$ns\_ at 0.0 "$Server2 color maroon"

## SETTING ANIMATION RATE

$ns\_ at 0.0 "$ns\_ set-animation-rate 12.5ms"

# COLORING THE NODES

$n9 color blue

$ns\_ at 4.71 "$n9 color blue"

$n5 color blue

$ns\_ at 7.0 "$n5 color blue"

$n2 color blue

$ns\_ at 7.29 "$n2 color blue"

$n16 color blue

$ns\_ at 7.59 "$n16 color blue"

$n9 color maroon

$ns\_ at 7.44 "$n9 color maroon"

$ns\_ at 7.43 "$n9 label TTLover"

$ns\_ at 7.55 "$n9 label \"\""

$n12 color blue

$ns\_ at 7.85 "$n12 color blue"

#### Establishing Communication

set udp0 [$ns\_ create-connection UDP $Server1 LossMonitor $n18 0]

$udp0 set fid\_ 1

set cbr0 [$udp0 attach-app Traffic/CBR]

$cbr0 set packetSize\_ 1000

$cbr0 set interopt\_ .07

$ns\_ at 0.0 "$cbr0 start"

$ns\_ at 4.0 "$cbr0 stop"

set udp1 [$ns\_ create-connection UDP $Server1 LossMonitor $n22 0]

$udp1 set fid\_ 1

set cbr1 [$udp1 attach-app Traffic/CBR]

$cbr1 set packetSize\_ 1000

$cbr1 set interopt\_ .07

$ns\_ at 0.1 "$cbr1 start"

$ns\_ at 4.1 "$cbr1 stop"

set udp2 [$ns\_ create-connection UDP $n21 LossMonitor $n20 0]

$udp2 set fid\_ 1

set cbr2 [$udp2 attach-app Traffic/CBR]

$cbr2 set packetSize\_ 1000

$cbr2 set interopt\_ .07

$ns\_ at 2.4 "$cbr2 start"

$ns\_ at 4.1 "$cbr2 stop"

set udp3 [$ns\_ create-connection UDP $Server1 LossMonitor $n15 0]

$udp3 set fid\_ 1

set cbr3 [$udp3 attach-app Traffic/CBR]

$cbr3 set packetSize\_ 1000

$cbr3 set interopt\_ 5

$ns\_ at 4.0 "$cbr3 start"

$ns\_ at 4.1 "$cbr3 stop"

set udp4 [$ns\_ create-connection UDP $Server1 LossMonitor $n14 0]

$udp4 set fid\_ 1

set cbr4 [$udp4 attach-app Traffic/CBR]

$cbr4 set packetSize\_ 1000

$cbr4 set interopt\_ 5

$ns\_ at 4.0 "$cbr4 start"

$ns\_ at 4.1 "$cbr4 stop"

set udp5 [$ns\_ create-connection UDP $n15 LossMonitor $n16 0]

$udp5 set fid\_ 1

set cbr5 [$udp5 attach-app Traffic/CBR]

$cbr5 set packetSize\_ 1000

$cbr5 set interopt\_ 5

$ns\_ at 4.0 "$cbr5 start"

$ns\_ at 4.1 "$cbr5 stop"

set udp6 [$ns\_ create-connection UDP $n15 LossMonitor $n17 0]

$udp6 set fid\_ 1

set cbr6 [$udp6 attach-app Traffic/CBR]

$cbr6 set packetSize\_ 1000

$cbr6 set interopt\_ 5

$ns\_ at 4.0 "$cbr6 start"

$ns\_ at 4.1 "$cbr6 stop"

set udp7 [$ns\_ create-connection UDP $n14 LossMonitor $n4 0]

$udp7 set fid\_ 1

set cbr7 [$udp7 attach-app Traffic/CBR]

$cbr7 set packetSize\_ 1000

$cbr7 set interopt\_ 5

$ns\_ at 4.0 "$cbr7 start"

$ns\_ at 4.1 "$cbr7 stop"

set udp8 [$ns\_ create-connection UDP $n14 LossMonitor $n9 0]

$udp8 set fid\_ 1

set cbr8 [$udp8 attach-app Traffic/CBR]

$cbr8 set packetSize\_ 1000

$cbr8 set interopt\_ 5

$ns\_ at 4.0 "$cbr8 start"

$ns\_ at 4.1 "$cbr8 stop"

set udp9 [$ns\_ create-connection UDP $n4 LossMonitor $n3 0]

$udp9 set fid\_ 1

set cbr9 [$udp9 attach-app Traffic/CBR]

$cbr9 set packetSize\_ 1000

$cbr9 set interopt\_ 5

$ns\_ at 4.0 "$cbr9 start"

$ns\_ at 4.1 "$cbr9 stop"

set udp10 [$ns\_ create-connection UDP $n4 LossMonitor $n2 0]

$udp10 set fid\_ 1

set cbr10 [$udp10 attach-app Traffic/CBR]

$cbr10 set packetSize\_ 1000

$cbr10 set interopt\_ 5

$ns\_ at 4.0 "$cbr10 start"

$ns\_ at 4.1 "$cbr10 stop"

set udp11 [$ns\_ create-connection UDP $n9 LossMonitor $n16 0]

$udp11 set fid\_ 1

set cbr11 [$udp11 attach-app Traffic/CBR]

$cbr11 set packetSize\_ 1000

$cbr11 set interopt\_ 5

$ns\_ at 4.0 "$cbr11 start"

$ns\_ at 4.1 "$cbr11 stop"

set udp12 [$ns\_ create-connection UDP $n9 LossMonitor $n10 0]

$udp12 set fid\_ 1

set cbr12 [$udp12 attach-app Traffic/CBR]

$cbr12 set packetSize\_ 1000

$cbr12 set interopt\_ 5

$ns\_ at 4.0 "$cbr12 start"

$ns\_ at 4.1 "$cbr12 stop"

#ANNOTATIONS DETAILS

$ns\_ at 0.0 "$ns\_ trace-annotate \"MOBILE NODE MOVEMENTS\""

$ns\_ at 4.1 "$ns\_ trace-annotate \"NODE27 CACHE THE DATA FRO SERVER\""

#$ns\_ at 4.59 "$ns\_ trace-annotate \"PACKET LOSS AT NODE27\""

$ns\_ at 4.71 "$ns\_ trace-annotate \"NODE10 CACHE THE DATA\""

### PROCEDURE TO STOP

proc stop {} {

global ns\_ tracefd

$ns\_ flush-trace

close $tracefd

exec nam aodv.nam &

exit 0

}

puts "Starting Simulation........"

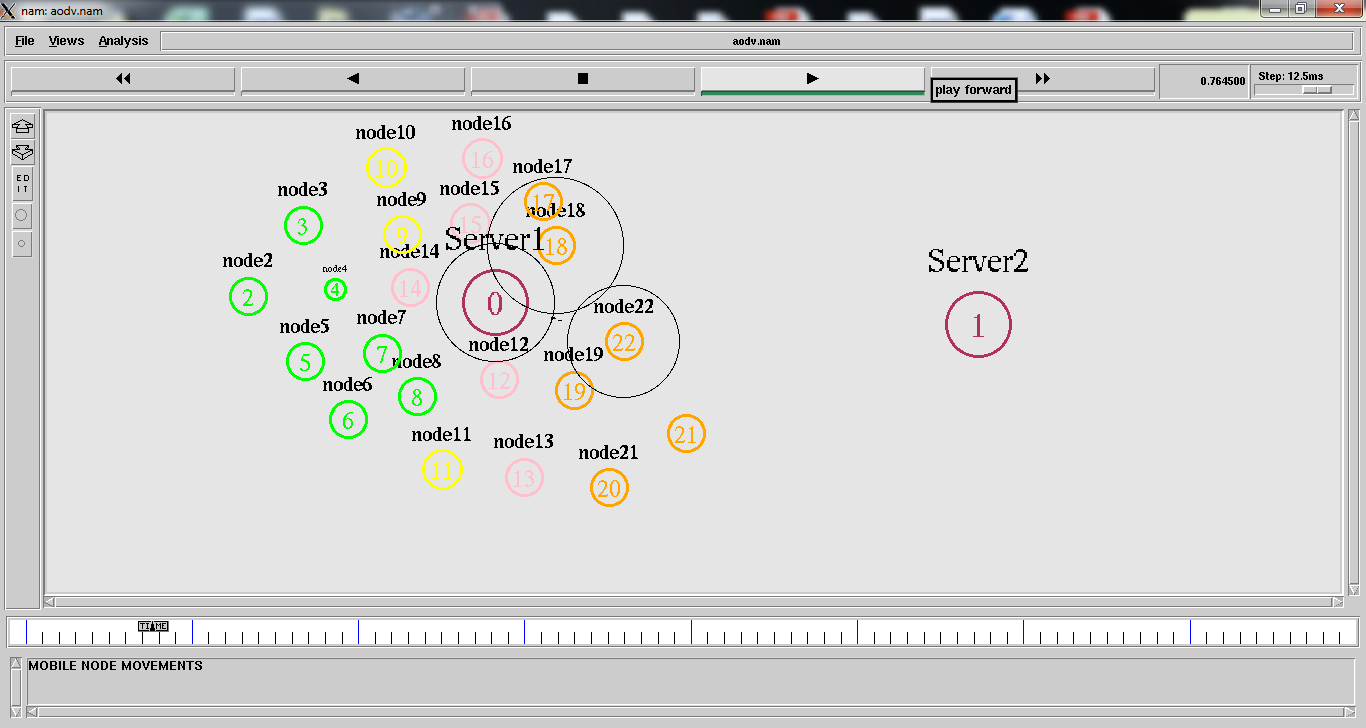
$ns\_ at 25.0 "stop"

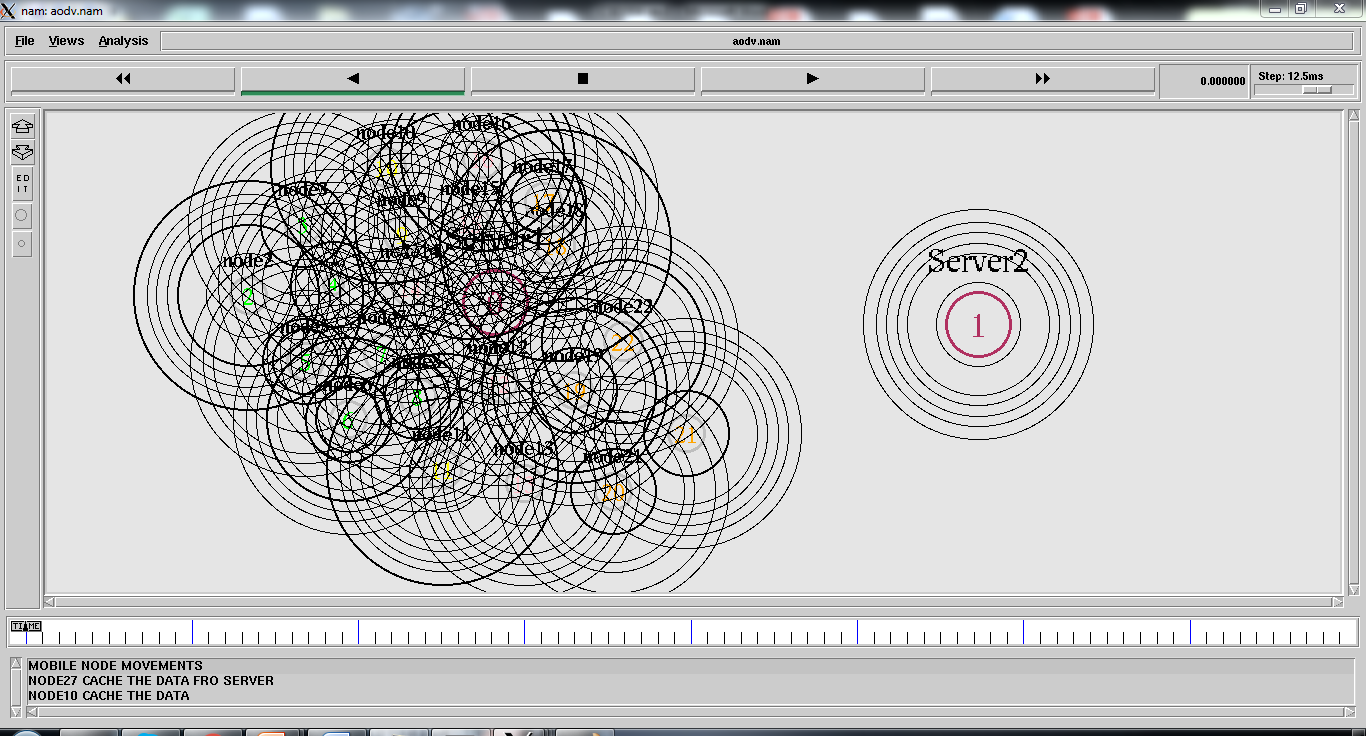
$ns\_ run

**CONCLUSION:**

As the objective of this research paper is to simulate the wireless network using network simulator NS2 and to measure the performance of this network using AODV as routing protocol. To measure the performance of AODV we select different performance matrix. The values are obtained from the trace file generated by NS2. Using the AWK script the data is processed and is used to measure the performance of AODV protocol.

**SNAPSHOT OF THE SIMULATED NETWORK:**



****

**RESULTS AND DISCUSSION:**

The AODV protocol is successfully implemented for MANET using ns2.It can clearly be seen that AODV has many practical applications in real life.For example, a variation of MANET called Vehicular Ad Hoc Network allows vehicles on roads to communicate within them.If any emergency vehicles like ambulance of fire engine is approaching,it can alert the vehicles ahead of the approaching vehicle so that people could make way for them.It can also be used in WPAN where a small number of devices are connected .This concept can also be used in foggy and dark areas so that vehicles are alerted of nearby vehicles thus avoiding accidents.This concept can also be used in aircrafts so that planes even though they fly through clouds could possibly know nearby aircrafts and thus it can prevent accidents.