EXPERIMENT NO. 4

AIM: Illustration of Hidden Terminal/Exposed terminal Problem using NS2/NS3.

THEORY:

A wireless network with lack of centralized control entity, sharing of wireless bandwidth among network access nodes i.e., medium access control (MAC) nodes must be organized in decentralized manner. The hidden terminal problem occurs when a terminal is visible from a wireless access point (APs), but not from other nodes communicating with that AP. This situation leads the difficulties in medium access control sublayer over wireless networking. In a formal way hidden terminal are nodes in a wireless network that are out of range of other node or a collection of nodes. Consider a wireless networking, each node at the far edge of the access point's range, which is known as A, can see the access point, but it is unlikely that the same node can see a node on the opposite end of the access point's range, C. These nodes are known as hidden. The problem is when nodes A and C start to send packets simultaneously to the access point B. Because the nodes A and C are out of range of each other and so cannot detect a collision while transmitting, Carrier sense multiple access with collision detection (CSMA/CD) does not work, and collisions occur, which then corrupt the data received by the access point. To overcome the hidden node problem, RTS/CTS handshaking (IEEE 802.11 RTS/CTS) is implemented in conjunction with the Carrier sense multiple accesses with collision avoidance (CSMA/CA) scheme. The same problem exists in a MANET. The transmission range of access point A reaches at B, but not at access point C, similarly transmission range of access point C reaches B, but not at A. These nodes are known as hidden terminals. The problem occurs when nodes A and C start to send data packets simultaneously to the access point B. Because the access points A and C are out of range of each other and resultant they cannot detect a collision while transmitting, Carrier sense multiple access with collision detection (CSMA/CD) does not work, and collisions occur, which then corrupt the data received by the access point B due to the hidden terminal problem.

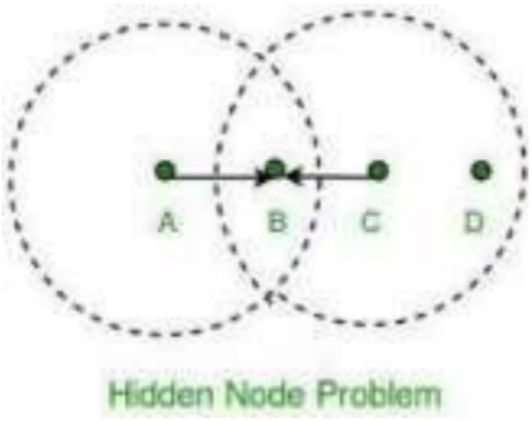
The hidden terminal analogy is described as follows:

• Terminal A sends data to B, terminal C cannot hear A

• Terminal C wants to send data to B, terminal C senses a "free" medium (CS fails) and starts transmitting

• Collision at B occurs, A cannot detect this collision (CD fails) and continues with its transmission to B

• Terminal A is "hidden" from C and vice versa.



The solution of hidden terminal problem is as follows.

When A wants to send a packet to B, A first sends a Request-to-send (RTS) to B. On receiving RTS, B responds by sending Clear-to-Send (CTS). When C overhears a CTS, it keeps quiet for the duration of the transfer. Transfer duration is included in both RTS and CTS. RTS and CTS are short frames, reduces collision chance.

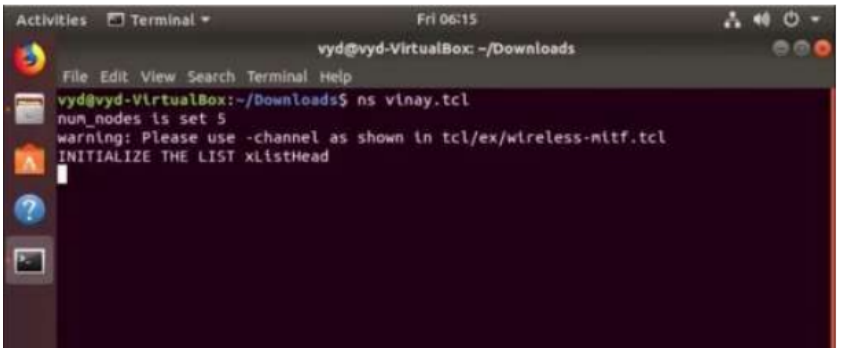
Program:

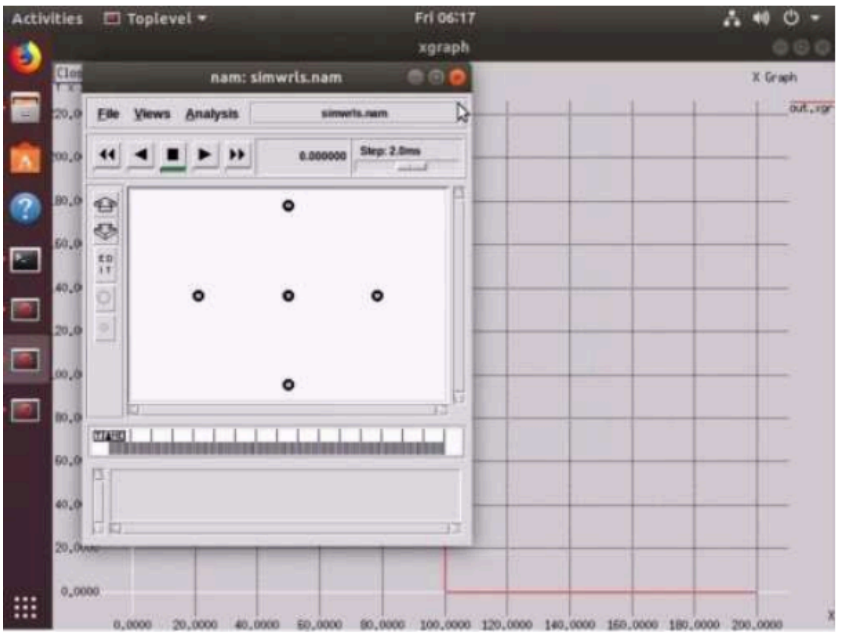
```
BEGIN{
sim_end = 200; i=0;
while (i<=sim_end) {sec[i]=0; i+=1;};
}
{
if ($1=="r" && $7=="cbr"&& $3=="_0_")
{ sec[int($2)]+=$8; };
}
END{ i=0;
while (i<=sim_end) {print i " " sec[i];
i+=1;};
}# Define options
set val(chan) Channel/WirelessChannel ;#
channel type set val(prop)
Propagation/FreeSpace ;#
radio-propagation model set val(netif)
Phy/WirelessPhy ;# network interface type
set val(mac)
Mac/802_11 ;# MAC type
set val(ifq) Queue/DropTail/PriQueue ;#
interface queue type set val(ll) LL ;# link
layer type
set val(ant) Antenna/OmniAntenna ;#
antenna model set val(ifqlen) 10000 ;# max
packet in
ifq set val(nn) 5 ;# number of mobilenodes
set val(rp) DSR ;# routing protocol
$node_(3) set X_ [expr $val(R) *2]
$node_(3) set Y_ $val(R)
$node_(3) set Z_ 0
$node_(4) set X_ $val(R)
$node_(4) set Y_ [expr $val(R) *2]
$node_(4) set Z_ 0
for {set i 0} {$i<$val(nn)} {incr i} {
$ns initial_node_pos $node_($i) 30
}
# Generation of movements
$ns at 0 "$node_(1) setdest $val(R) $val(R) 3.0"
$ns at 0 "$node_(2) setdest $val(R) $val(R) 3.0"
$ns at 0 "$node_(3) setdest $val(R) $val(R) 3.0"
$ns at 0 "$node_(4) setdest $val(R) $val(R) 3.0"
# Set a TCP connection between node_(0) and
node_(1) set tcp [new Agent/TCP/Newreno]
#$tcp set class_ 2
set tcp [new Agent/UDP]
$tcp set class_ 2
set sink [new Agent/Null]
$ns attach-agent $node_(1) $tcp
$ns attach-agent $node_(0) $sink
$ns connect $tcp $sink
set ftp [new Application/Traffic/CBR]
$ftp attach-agent $tcp
$ns at 0.0 "$ftp start"
```

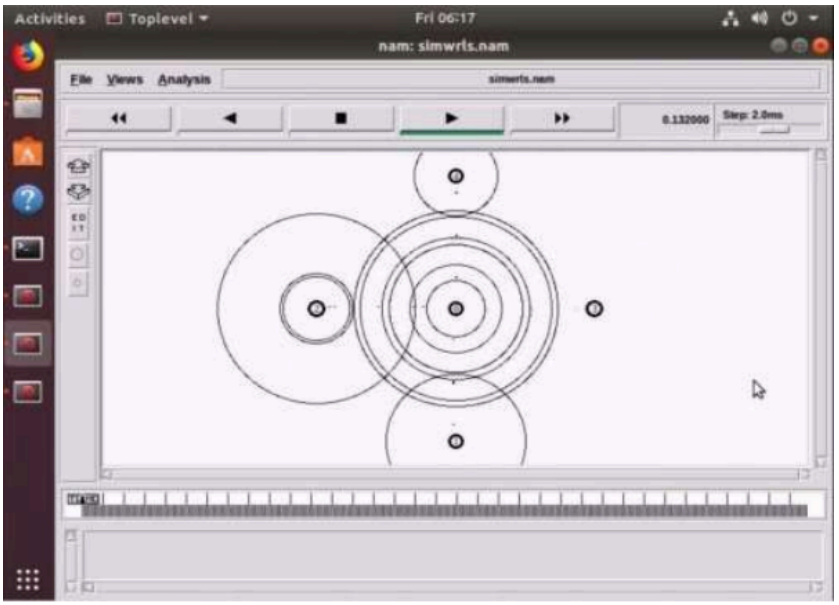
```
set val(x) 600 ;# X dimension of topography set val(y) 600
;# Y dimension of topography set val(stop) 100 ;# time of simulation end
set val(R) 300
set opt(tr) out.tr
set ns [new Simulator] set tracefd [open $opt(tr) w] set windowVsTime2 [open win.tr w]
set namtrace [open simwrls.nam w] Mac/802_11 set dataRate_ 1.2e6 Mac/802_11 set RTSThreshold_ 100 $ns
trace-all $tracefd #$ns use- newtrace
$ns namtrace-all-wireless $namtrace $val(x) $val(y)
# set up topography object set topo [new Topography]
$topo load_flatgrid $val(x) $val(y) create-god $val(nn)
#
# Create nn mobilenodes [$val(nn)] and attach them to the channel. # # configure the nodes
$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) \
-channelType $val(chan) \
-topoInstance $topo \
-agentTrace ON \
-routerTrace ON \
-macTrace ON \
-movementTrace ON Phy/WirelessPhy set CSThresh 30.5e-10
for {set i 0} {$i < $val(nn) } { incr i } { set node_($i) [$ns node] }
$node_(0) set X_ $val(R)
$node_(0) set Y_ $val(R)
$node_(0) set Z_ 0
$node_(1) set X_ $val(R)
$node_(1) set Y_ 0
$node_(1) set Z_ 0
$node_(2) set X_ 0
$node_(2) set Y_ $val(R)
$node_(2) set Z_ 0
#
##########################################
######
 For coloring but doesnot work
##########################################
#######
$tcp set fid_ 1
$ns color 1 blue #////////////////////////////////////////////
set tcp [new Agent/UDP]
$tcp set class_ 2
set sink [new Agent/Null]
$ns attach-agent $node_(2) $tcp
$ns attach-agent $node_(0) $sink
$ns connect $tcp $sink
set ftp [new Application/Traffic/CBR]
$ftp attach-agent $tcp
$ns at 0.0 "$ftp start" set tcp [new Agent/UDP]
$tcp set class_ 2
set sink [new Agent/Null]
$ns attach-agent $node_(3) $tcp
$ns attach-agent $node_(0) $sink
$ns connect $tcp $sink
set ftp [new Application/Traffic/CBR]
$ftp attach-agent $tcp
$ns at 0.0 "$ftp start" set tcp [new Agent/UDP]
$tcp set class_ 2
set sink [new Agent/Null]
$ns attach-agent $node_(4) $tcp
$ns attach-agent $node_(0) $sink
$ns connect $tcp $sink
set ftp [new Application/Traffic/CBR]
$ftp attach-agent $tcp
$ns at 0.0 "$ftp start"
# Telling nodes when the simulation ends #for {set i 0}
 {$i < $val(nn) } { incr i } { # $ns at $val(stop) "$node_($i) reset"; #}
# ending nam and the simulation
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"
$ns at $val(stop) "stop"
$ns at $val(stop) "puts \"end simulation\" ; $ns halt"
proc stop {} { exec awk -f fil.awk out.tr
> out.xgr exec xgraph out.xgr &
global ns tracefd namtrace
$ns flush-trace close $tracefd close $namtrace exec nam simwrls.nam &
}
$ns run
```

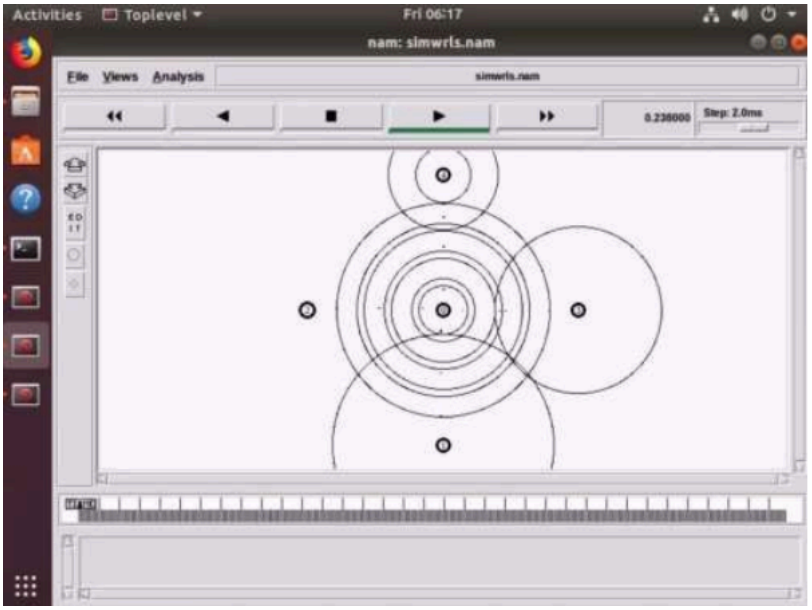
Output:

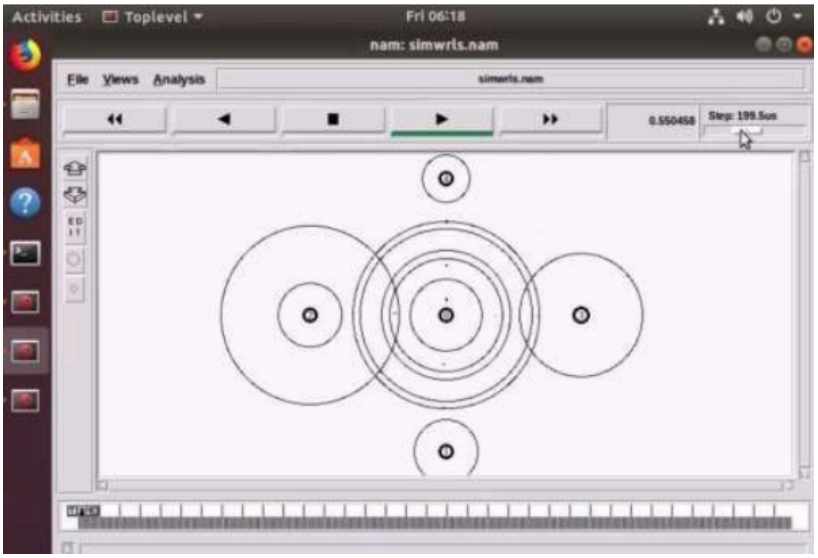
1) The node 0 and 2 want to send data to node 1 the range of node 0 and 2 is limited to 1 they do not know that other node is also sending data to 1 and therefore collision occurs.






Activities
Toplevel
Fri 06:17
nam: simwrls.nam
File
Views
Analysis
simwrls.nam
0.132000
Step: 2.0ms


Activities
Toplevel
Fri 06:17
nam: simwrls.nam
File
Views
Analysis
simwrls.nam
0.238000
Step: 2.0ms


Activities
Toplevel
Fri 06:18
nam: simwrls.nam
File
Views
Analysis


Activities
Unknown
Fri 06:18
xgraph
Close
Hdcpy
About
Anim
Replot
Servr
X Graph