```
# This script is created by NSG2 beta1
# <http://wushoupong.googlepages.com/nsg>

#===================================
#     Simulation parameters setup
#===================================
set val(stop)   10.0                         ;# time of simulation end

#===================================
#        Initialization
#===================================
#Create a ns simulator
set ns [new Simulator]

#Open the NS trace file
set tracefile [open out.tr w]
$ns trace-all $tracefile

#Open the NAM trace file
set namfile [open out.nam w]
$ns namtrace-all $namfile

#===================================
#        Nodes Definition
#===================================
#Create 3 nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]

#===================================
#        Links Definition
#===================================
#Createlinks between nodes
$ns duplex-link $n0 $n1 10.0Mb 10ms DropTail
$ns queue-limit $n0 $n1 5
$ns duplex-link $n2 $n1 12.0Mb 12ms DropTail
$ns queue-limit $n2 $n1 12

#Give node position (for NAM)
$ns duplex-link-op $n0 $n1 orient right-down
$ns duplex-link-op $n2 $n1 orient right-up

#===================================
#        Agents Definition
#===================================
#Setup a TCP connection
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0
```

```
set sink2 [new Agent/TCPSink]
$ns attach-agent $n1 $sink2
$ns connect $tcp0 $sink2
$tcp0 set packetSize_ 1500

#Setup a TCP connection
set tcp1 [new Agent/TCP]
$ns attach-agent $n2 $tcp1
set sink3 [new Agent/TCPSink]
$ns attach-agent $n1 $sink3
$ns connect $tcp1 $sink3
$tcp1 set packetSize_ 1500


#===================================
#        Applications Definition
#===================================
#Setup a FTP Application over TCP connection
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
$ns at 0.2 "$ftp0 start"
$ns at 2.0 "$ftp0 stop"

#Setup a FTP Application over TCP connection
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
$ns at 0.5 "$ftp1 start"
$ns at 3.0 "$ftp1 stop"


#===================================
#        Termination
#===================================
#Define a 'finish' procedure
proc finish {} {
    global ns tracefile namfile
    $ns flush-trace
    close $tracefile
    close $namfile
    exec nam out.nam &
    exit 0
}
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"
$ns at $val(stop) "finish"
$ns at $val(stop) "puts \"done\" ; $ns halt"
$ns run
```

```
# This script is created by NSG2 beta1
# <http://wushoupong.googlepages.com/nsg>

#===================================
#     Simulation parameters setup
#===================================
set val(stop)   10.0                         ;# time of simulation end

#===================================
#        Initialization
#===================================
#Create a ns simulator
set ns [new Simulator]

#Open the NS trace file
set tracefile [open out.tr w]
$ns trace-all $tracefile

#Open the NAM trace file
set namfile [open out.nam w]
$ns namtrace-all $namfile

#===================================
#        Nodes Definition
#===================================
#Create 6 nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
$n0 label "ping0"
$n1 label "ping1"
$n2 label "R1"
$n3 label "R2"
$n4 label "ping4"
$n5 label "ping5"
$ns color 1 red
$ns color 2 blue
$ns color 3 green
$ns color 4 orange

#===================================
#        Links Definition
#===================================
#Createlinks between nodes
$ns duplex-link $n0 $n2 0.4Mb 10ms DropTail
```

```
$ns duplex-link $n1 $n2 1Mb 10ms DropTail

$ns duplex-link $n2 $n3 4Kb 10ms DropTail

$ns duplex-link $n3 $n4 1Mb 10ms DropTail

$ns duplex-link $n3 $n5 1Mb 10ms DropTail


#Give node position (for NAM)
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right
$ns duplex-link-op $n3 $n4 orient right-up
$ns duplex-link-op $n3 $n5 orient right-down

set ping0 [new Agent/Ping]
$ns attach-agent $n0 $ping0

set ping1 [new Agent/Ping]
$ns attach-agent $n1 $ping1

set ping4 [new Agent/Ping]
$ns attach-agent $n4 $ping4

set ping5 [new Agent/Ping]
$ns attach-agent $n5 $ping5

$ns connect $ping0 $ping4
$ns connect $ping1 $ping5
proc sendPingPacket {} {
        global ns ping0 ping1
        set intervalTime 0.001
        set now [$ns now]
        $ns at [expr $now + $intervalTime] "$ping0 send"
        $ns at [expr $now + $intervalTime] "$ping1 send"
        $ns at [expr $now + $intervalTime] "sendPingPacket"
}
#rtt=round trip time(packet travel from src to dest and back to src)
Agent/Ping instproc recv {from rtt} {
        global seq
        $self instvar node_
        puts "The node [$node_ id] received an ACK from the node $from with RTT $rtt ms"
}
$ping0 set class_ 1
$ping1 set class_ 2
$ping4 set class_ 3
$ping5 set class_ 4
```

```
#====================================
#       Termination
#====================================
#Define a 'finish' procedure
proc finish {} {
    global ns tracefile namfile
    $ns flush-trace
    close $tracefile
    close $namfile
    exec nam out.nam &
    exit 0
}
$ns at 0.01 "sendPingPacket"
$ns at 10.0 "finish"
$ns run
```

```
set val(stop) 10.0      ;# time of simulation end
#===================================
#     Initialization
#===================================
#Create a ns simulator
set ns [new Simulator]
#add manually

#end
#Open the NS trace file
set tracefile [open 5.tr w]
$ns trace-all $tracefile
#Open the NAM trace file
set namfile [open 5.nam w]
$ns namtrace-all $namfile
#add manually
set wf0 [open WinFile0 w]
set wf1 [open WinFile1 w]
proc PlotWindow {tcpSource file} {
global ns
set time 0.1
set now [$ns now]
set cwnd [$tcpSource set cwnd_]
puts $file "$now $cwnd"
$ns at [expr $now+$time] "PlotWindow $tcpSource $file"
}
#end
#===================================
#     Nodes Definition
#===================================
#Create 6 nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
$n0 label "Source0"
$n1 label "Source1"
$n2 label "R1"
$n3 label "R2"
$n4 label "Dest0"
$n5 label "Dest1"
$ns color 1 "red"
$ns color 2 "green"
$ns color 3 "blue"
$ns color 4  "orange"

#===================================
#     Links Definition
#===================================

#add manually
set lan [$ns newLan "$n0 $n1 $n2" 0.5Mb 40ms LL Queue/DropTail MAC/802_3 Channel]
$ns duplex-link $n2 $n3 10Mb 100ms DropTail
$ns duplex-link-op $n2 $n3 queuePos 0.5
set lan [$ns newLan "$n3 $n4 $n5" 0.5Mb 40ms LL Queue/DropTail MAC/802_3 Channel]
set loss_module [new ErrorModel]
$loss_module ranvar [new RandomVariable/Uniform]
$loss_module drop-target [new Agent/Null]
$ns lossmodel $loss_module $n2 $n3

#end
#===================================
#     Agents Definition
#===================================
#Setup a TCP connection
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0
set sink2 [new Agent/TCPSink]
$ns attach-agent $n4 $sink2
$ns connect $tcp0 $sink2
$tcp0 set packetSize_ 1500
#Setup a TCP connection
set tcp1 [new Agent/TCP]
$ns attach-agent $n1 $tcp1
set sink3 [new Agent/TCPSink]
$ns attach-agent $n5 $sink3
$ns connect $tcp1 $sink3
$tcp1 set packetSize_ 1500
#===================================
#     Applications Definition
#===================================
#Setup a FTP Application over TCP connection
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
$ns at 0.1 "$ftp0 start"
$ns at 9.8 "$ftp0 stop"
#Setup a FTP Application over TCP connection
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
$ns at 1 "$ftp1 start"
$ns at 9.9 "$ftp1 stop"
#add manually
$ns at 0.1 "PlotWindow $tcp0 $wf0"
$ns at 0.5 "PlotWindow $tcp1 $wf1"
$tcp0 set class_ 1
$tcp1 set class_ 2
```

```
#end
#===================================
#     Termination
#===================================
#Define a 'finish' procedure
proc finish {} {
  global ns tracefile namfile
  $ns flush-trace
  close $tracefile
  close $namfile
  exec nam 5.nam &
  exec xgraph WinFile0 WinFile1 &
  exit 0
}
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"
$ns at $val(stop) "finish"
$ns at $val(stop) "puts \"done\" ; $ns halt"
$ns run
```

```
# This script is created by NSG2 beta1
# <http://wushoupong.googlepages.com/nsg>
if {$argc != 1} {

        exit 0
}
#===================================
#     Simulation parameters setup
#===================================
set val(chan)   Channel/WirelessChannel    ;# channel type
set val(prop)   Propagation/TwoRayGround   ;# 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) 50                         ;# max packet in ifq
set val(nn)     [lindex $argv 0]           ;# number of mobilenodes
set val(rp)     AODV                       ;# routing protocol
set val(x)      750                        ;# X dimension of topography
set val(y)      750                        ;# Y dimension of topography
set val(stop)   100.0                      ;# time of simulation end

#===================================
#        Initialization
#===================================
#Create a ns simulator
set ns [new Simulator]

#Setup topography object
set topo       [new Topography]
$topo load_flatgrid $val(x) $val(y)
create-god $val(nn)

#Open the NS trace file
set tracefile [open out.tr w]
$ns trace-all $tracefile

#Open the NAM trace file
set namfile [open out.nam w]
$ns namtrace-all $namfile
$ns namtrace-all-wireless $namfile $val(x) $val(y)
set chan [new $val(chan)];#Create wireless channel

#===================================
#     Mobile node parameter setup
#===================================
$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      $chan \
                -topoInstance  $topo \
                -agentTrace    ON \
                -routerTrace   ON \
                -macTrace      OFF \
                -movementTrace OFF

#===================================
#       Nodes Definition
#===================================
#add manually
for {set i 0} {$i < $val(nn)} {incr i} {
        set n($i) [$ns node]
}

#Randomly placing the nodes
for {set i 0} {$i < $val(nn)} {incr i} {
        set XX [expr rand()*750]
        set YY [expr rand()*750]
        $n($i) set X_ $XX
        $n($i) set Y_ $YY
}

$ns at 0.0 "destination"

for {set i 0} {$i < $val(nn)} {incr i} {
        $ns initial_node_pos $n($i) 50
}

proc destination {} {
        global ns val n
        set now [$ns now]
        set time 3.0
        for {set i 0} {$i < $val(nn)} {incr i} {
                set XX [expr rand()*750]
                set YY [expr rand()*750]
                $ns at [expr $now + $time] "$n($i) setdest $XX $YY 20.0"
        }
        $ns at [expr $now + $time] "destination"
}

#end
#===================================
#       Agents Definition
#===================================
#add manually
#Setup a TCP connection
set tcp0 [new Agent/TCP]
$ns attach-agent $n(0) $tcp0
set sink1 [new Agent/TCPSink]
$ns attach-agent $n(5) $sink1
$ns connect $tcp0 $sink1
$tcp0 set packetSize_ 1500
#end

#===================================
#       Applications Definition
#===================================
#Setup a FTP Application over TCP connection
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
$ns at 1.0 "$ftp0 start"



#===================================
#       Termination
#===================================
#Define a 'finish' procedure
proc finish {} {
    global ns tracefile namfile
    $ns flush-trace
    close $tracefile
    close $namfile
    exec nam out.nam &
    exec awk -f 4.awk out.tr &
    exit 0
}
for {set i 0} {$i < $val(nn) } { incr i } {
    $ns at $val(stop) "$n($i) reset"
}
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"
$ns at $val(stop) "finish"
$ns at $val(stop) "puts \"done\" ; $ns halt"
$ns run
```

```
EGIN{
        PacketRcvd=0;
        Throughput=0.0;
}
{
        if(($1=="r")&&($3=="_5_")&&($4=="AGT")&&($7=="tcp")&&($8>1000))
        {
                PacketRcvd++;
        }
}
END {
        Throughput=((PacketRcvd*1500*8)/(99.0*1000000));
        printf "the throughput is:%f\n",Throughput;
}
```

```
set val(chan)            Channel/WirelessChannel
set val(type)            GSM
set val(prop)            Propagation/TwoRayGround
set val(netif)           Phy/WirelessPhy
set val(mac)             Mac/802_11
set val(ifq)             Queue/DropTail/PriQueue
set val(ll)              LL
set val(ant)             Antenna/OmniAntenna
set val(x)                1500
set val(y)                1500
set val(ifqlen)           1000
set val(adhocRouting)   AODV
set val(nn)              10
set val(stop)            5.0

set f0 [open out02.tr w]
set f1 [open lost02.tr w]
set f2 [open delay02.tr w]

set ns_                  [new Simulator]
set topo                 [new Topography]

set tracefd              [open out.tr w]
set namtrace             [open out.nam w]

$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)

$topo load_flatgrid $val(x) $val(y)

set god_ [create-god $val(nn)]
$ns_ color 0 red
$ns_ node-config -adhocRouting AODV \
          -llType $val(ll) \
          -macType $val(mac) \
          -ifqType $val(ifq) \
          -ifqLen $val(ifqlen) \
          -antType $val(ant) \
          -propType $val(prop) \
          -phyType $val(netif) \
          -channelType $val(chan) \
              -energyModel EnergyModel \
              -initialEnergy 100 \
               -rxPower 0.3 \
               -txPower 0.6 \
               -topoInstance $topo \
          -agentTrace ON \
          -routerTrace ON \
          -macTrace OFF
```

```
for {set i 0} {$i < $val(nn) } {incr i} {
            set node_($i) [$ns_ node]

}


 set X1(0) 1035.201
 set Y1(0) 444.699
 set X1(1) 244.365
 set Y1(1) 521.418
 set X1(2) -18.1268
 set Y1(2) 300.612
 set X1(3) 723.89
 set Y1(3) 343.533
 set X1(4) 122.34
 set Y1(4) 311.755
 set X1(5) 373.498
 set Y1(5) 472.206
 set X1(6) 548.549
 set Y1(6) 361.062
 set X1(7) 389.995
 set Y1(7) 381.178
 set X1(8) 494.798
 set Y1(8) 477.771
 set X1(9) 275.01
 set Y1(9) 381.99


for {set i 0} {$i < $val(nn) } {incr i} {
            $node_($i) set X_ $X1($i)
        $node_($i) set Y_ $Y1($i)
        $node_($i) set Z_ 0.0


}

puts "---------------------------------------"
set m 0
puts "----------------------------------------"
puts "|    Node      |  One hop neighbour    |"
puts "----------------------------------------"
for {set i 0} {$i < $val(nn) } {incr i} {

set k 0
for {set j 0} {$j < $val(nn) } {incr j} {
```

```
set a [ expr $X1($j)-$X1($i)]
set b [ expr $a*$a]
set c [ expr $Y1($j)-$Y1($i)]
set d [ expr $c*$c]
set e [ expr $b+$d]
set f 0.5
set g [expr pow($e,$f)]
#puts "Distance from node($i) --to--node($j)----------->$g"
if {$g <= 200 && $i != $j} {

puts "|    node($i)     |     node($j)       |"

set nei($m) $j

set k [expr $k+1]
set m [ expr $m+1]
}

}
puts "----------------------------------------"
}


 puts "Loading connection pattern..."


puts "Loading scenario file..."
for {set i 0} {$i < $val(nn) } {incr i} {

    $ns_ initial_node_pos $node_($i) 45
}

for {set i 0} {$i < $val(nn) } {incr i} {
    $ns_ at $val(stop).0 "$node_($i) reset";
}

set udp_(0) [new Agent/UDP]
$ns_ attach-agent $node_(2) $udp_(0)
set sink [new Agent/LossMonitor]
$ns_ attach-agent $node_(3) $sink
set cbr1_(0) [new Application/Traffic/CBR]
$cbr1_(0) set packetSize_ 1000
$cbr1_(0) set interval_ 0.1
$cbr1_(0) set maxpkts_ 1000
$cbr1_(0) attach-agent $udp_(0)
$ns_ connect $udp_(0) $sink
$ns_ at 1.00 "$cbr1_(0) start"
```

```
set holdtime 0
set holdseq 0

set holdrate1 0

proc record {} {
global sink  f0 f1 f2 holdtime holdseq holdrate1

set ns [Simulator instance]
set time 0.9 ;#Set Sampling Time to 0.9 Sec

set bw0 [$sink set bytes_]
set bw1 [$sink set nlost_]

set bw2 [$sink set lastPktTime_]
set bw3 [$sink set npkts_]

set now [$ns now]

        # Record Bit Rate in Trace Files
        puts $f0 "$now [expr (($bw0+$holdrate1)*8)/(2*$time*1000000)]"


        # Record Packet Loss Rate in File
        puts $f1 "$now [expr $bw1/$time]"

if { $bw3 > $holdseq } {
                puts $f2 "$now [expr ($bw2 - $holdtime)/($bw3 - $holdseq)]"
        } else {
                puts $f2 "$now [expr ($bw3 - $holdseq)]"
        }

$sink set bytes_ 0
$sink set nlost_ 0

set holdtime $bw2
set holdseq $bw3

set  holdrate1 $bw0
    $ns at [expr $now+$time] "record"   ;# Schedule Record after $time interval sec
}


# Start Recording at Time 0
$ns_ at 0.0 "record"

source link1.tcl
```

```
proc stop {} {
        global ns_ tracefd f0 f1 f2

        # Close Trace Files
        close $f0
        close $f1
        close $f2
        exec nam out.nam
 # Plot Recorded Statistics

        exec xgraph out02.tr -geometry -x TIME -y thr -t Throughput 800x400 &
        exec xgraph lost02.tr  -geometry -x TIME -y loss -t Packet_loss 800x400 &
        exec xgraph delay02.tr  -geometry -x TIME -y delay -t End-to-End-Delay 800x400 &

$ns_ flush-trace

}

$ns_ at $val(stop) "stop"
$ns_ at  $val(stop).0002 "puts \"NS EXITING...\" ; $ns_ halt"
puts $tracefd "M 0.0 nn $val(nn) x $val(x) y $val(y) rp "
puts $tracefd "M 0.0 prop $val(prop) ant $val(ant)"

puts "Starting Simulation..."
$ns_ run
```

```
$ns_ at 0.1 "$node_(0) setdest 786 813 20"
$ns_ at 0.1 "$node_(1) setdest 895 890 20"
$ns_ at 0.1 "$node_(2) setdest 633 669 20"
$ns_ at 0.1 "$node_(3) setdest 1375 712 20"
$ns_ at 0.1 "$node_(4) setdest 773 680 20"
$ns_ at 0.1 "$node_(5) setdest 1024 841 20"
$ns_ at 0.1 "$node_(6) setdest 1199 730 20"
$ns_ at 0.1 "$node_(7) setdest 1041 750 20"
$ns_ at 0.1 "$node_(8) setdest 1146 846 20"
$ns_ at 0.1 "$node_(9) setdest 926 751 20"



$ns_ at 0.5 "$node_(2) add-mark m blue square"
$ns_ at 0.5 "$node_(3) add-mark m blue square"

$ns_ at 0.5 "$node_(2) label source"
$ns_ at 0.5 "$node_(3) label Destination"
```

```
puts "Enter number of nodes"
set tnn [gets stdin]
set val(chan)           Channel/WirelessChannel
set val(prop)           Propagation/TwoRayGround
set val(netif)          Phy/WirelessPhy
set val(mac)            Mac/802_11
set val(ifq)            Queue/DropTail/PriQueue
set val(ll)     LL
set val(ant)            Antenna/OmniAntenna
set val(x)          1500 #add manually
set val(y)          1500 #add manually
set val(ifqlen)       1000#add manually
set val(adhocRouting)   AODV #add manually
set val(nn)          $tnn
set val(stop)           10.0
#add manually
Mac/802_11 set cdma_code_bw_start_              0     ;# cdma code for bw request (start)
Mac/802_11 set cdma_code_bw_stop_               63     ;# cdma code for bw request (stop)
Mac/802_11 set cdma_code_init_start_            64     ;# cdma code for initial request (start)
Mac/802_11 set cdma_code_init_stop_             127     ;# cdma code for initial request (stop)
Mac/802_11 set cdma_code_cqich_start_    128    ;# cdma code for cqich request (start)
Mac/802_11 set cdma_code_cqich_stop_            195     ;# cdma code for cqich request (stop)
Mac/802_11 set cdma_code_handover_start_        196    ;# cdma code for handover request (start)
Mac/802_11 set cdma_code_handover_stop_         255     ;# cdma code for handover request (stop)
#end
set f0 [open out02.tr w]
set f1 [open lost02.tr w]
set f2 [open delay02.tr w]


set ns_                 [new Simulator]
set topo                [new Topography]

set tracefd             [open out.tr w]
set namtrace            [open out.nam w]

$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)

$topo load_flatgrid $val(x) $val(y)

set god_ [create-god $val(nn)]
$ns_ color 0 red
$ns_ node-config -adhocRouting AODV \
            -llType $val(ll) \
            -macType $val(mac) \
            -ifqType $val(ifq) \
            -ifqLen $val(ifqlen) \
            -antType $val(ant) \
              -propType $val(prop) \
              -phyType $val(netif) \
              -channelType $val(chan) \
                    -energyModel EnergyModel \
                    -initialEnergy 100 \
                     -rxPower 0.3 \
                     -txPower 0.6 \
                     -topoInstance $topo \
              -agentTrace ON \
              -routerTrace ON \
              -macTrace OFF



#add manually
for {set i 0} {$i < $val(nn) } {incr i} {
          set node_($i) [$ns_ node]
$node_($i) set X_  [expr rand() * 500]
$node_($i) set Y_ [expr rand() * 500]
$node_($i) set Z_ 0.000000000000;
}

for {set i 0} {$i < $val(nn) } {incr i} {
set xx  [expr rand() * 1500]
set yy [expr rand() * 1000]
$ns_ at 0.1 "$node_($i) setdest $xx 4yy 5"
}

 puts "Loading connection pattern..."


puts "Loading scenario file..."
for {set i 0} {$i < $val(nn) } {incr i} {

    $ns_ initial_node_pos $node_($i) 55
}

for {set i 0} {$i < $val(nn) } {incr i} {
    $ns_ at $val(stop).0 "$node_($i) reset";
}

puts "Enter source node"
set source [gets stdin]
puts "Enter destination node"
set dest [gets stdin]

set udp_(0) [new Agent/UDP]
$ns_ attach-agent $node_($source) $udp_(0)
set sink [new Agent/LossMonitor]
```

```
$ns_ attach-agent $node_($dest) $sink
set cbr1_(0) [new Application/Traffic/CBR]
$cbr1_(0) set packetSize_ 1000
$cbr1_(0) set interval_ 0.1
$cbr1_(0) set maxpkts_ 10000
$cbr1_(0) attach-agent $udp_(0)
$ns_ connect $udp_(0) $sink
$ns_ at 1.00 "$cbr1_(0) start"

set holdtime 0
set holdseq 0

set holdrate1 0

proc record {} {
global sink  f0 f1 f2 holdtime holdseq holdrate1

set ns [Simulator instance]
set time 0.9 ;#Set Sampling Time to 0.9 Sec

set bw0 [$sink set bytes_]
set bw1 [$sink set nlost_]

set bw2 [$sink set lastPktTime_]
set bw3 [$sink set npkts_]

set now [$ns now]

        # Record Bit Rate in Trace Files
        puts $f0 "$now [expr (($bw0+$holdrate1)*8)/(2*$time*1000000)]"


        # Record Packet Loss Rate in File
        puts $f1 "$now [expr $bw1/$time]"

if { $bw3 > $holdseq } {
                puts $f2 "$now [expr ($bw2 - $holdtime)/($bw3 - $holdseq)]"
        } else {
                puts $f2 "$now [expr ($bw3 - $holdseq)]"
        }

$sink set bytes_ 0
$sink set nlost_ 0

set holdtime $bw2
set holdseq $bw3

set  holdrate1 $bw0
    $ns at [expr $now+$time] "record"   ;# Schedule Record after $time interval sec
}
#end

# Start Recording at Time 0
$ns_ at 0.0 "record"

source link.tcl

proc stop {} {
        global ns_ tracefd f0 f1 f2

        # Close Trace Files
        close $f0
        close $f1
        close $f2
        exec nam out.nam
        exec xgraph out02.tr -geometry -x TIME -y thr -t Throughput 800x400 &
        exec xgraph lost02.tr  -geometry -x TIME -y loss -t Packet_loss 800x400 &
        exec xgraph delay02.tr  -geometry -x TIME -y delay -t End-to-End-Delay 800x400 &

$ns_ flush-trace

}

$ns_ at $val(stop) "stop"
$ns_ at  $val(stop).0002 "puts \"NS EXITING...\" ; $ns_ halt"
puts $tracefd "M 0.0 nn $val(nn) x $val(x) y $val(y) rp "
puts $tracefd "M 0.0 prop $val(prop) ant $val(ant)"

puts "Starting Simulation..."
$ns_ run
```

```
$ns_ at 0.5 "$node_($source) add-mark m blue square"
$ns_ at 0.5 "$node_($dest) add-mark m magenta square"

$ns_ at 0.5 "$node_($source) label SENDER"
$ns_ at 0.5 "$node_($dest) label RECEIVER"


$ns_ at 0.01 "$ns_ trace-annotate \"Network Deployment\""
```