ДОДАТОК А

Програмний код сценарію моделювання

# Параметри моделі

#

set val(chan) Channel/WirelessChannel ;# клас каналу

set val(prop) Propagation/TwoRayGround ;# модель розповсюдження сигналу

set val(netif) Phy/WirelessPhy/802\_15\_4 ;# клас мережевого інтерфейсу

set val(mac) Mac/802\_15\_4 ;# клас MAC

set val(ifq) Queue/DropTail/PriQueue ;# клас черги інтерфейсу

set val(ll) LL/LL802\_15\_4 ;# клас рівню LL

set val(ant) Antenna/OmniAntenna ;# клас антени

set val(rp) NOAH ;# клас агенту маршрутизації

set val(ifqlen) 3 ;# розмір черги інтерфейсу

set val(x) 10 ;# X розмір поля розташування

set val(y) 10 ;# Y розмір поля розташування

set val(assocStart) 0.6 ;# час початку асоціації

set val(assocTime) 1.3 ;# час асоціації одного вузл

set val(beacon\_enabled) 1 ;# включити режим beacon-enabled

set val(BO) "5" ;# beacon order

set val(SO) "5" ;# superframe order

set val(GTS\_setting) 0x8E ;# параметри GTS: 8 слотів

Agent/NOAH set be\_random\_ 0 ;# відключити jitter NOAH

# Параметри командної строки

set argNN [lindex $argv 0] ;# кількість вузлів

set argSizeBack [lindex $argv 1] ;# розмір пакету від Клієнта

set argIntervalBack [lindex $argv 2] ;# інтервал пакетів від Клієнта

set argSize [lindex $argv 3] ;# розмір пакету від Сервера

set val(nn) [expr {$argNN != "" ? $argNN : 2}] ;# кількість вузлів >= 2

# Час початку та закінчення відправки трафіка

set val(operationStart) [expr $val(assocStart) + $val(assocTime) \* $val(nn)]

set val(stop) [expr $val(operationStart) + 10.1]

# Параметри трафіка від Сервера до Клієнтів

set val(bmsg-interval) 0.012 ;# інтервал, с

set val(bmsg-size) [expr {$argSize != "" ? $argSize : 120}] ;# розмір, байт

set val(bmsg-start) $val(operationStart) ;# час початку

set val(bmsg-stop) [expr $val(stop) - 0.1] ;# час останову

# Параметри трафіка від Клієнтів до Сервера

set val(pois-interval) [expr {$argIntervalBack != "" ? $argIntervalBack : 3}]

set val(pois-size) [expr {$argSizeBack != "" ? $argSizeBack : 10}]

set val(pois-rate) 250

set val(pois-start) $val(operationStart)

set val(pois-stop) [expr $val(stop) - 0.1]

#

# Створення моделі

#

set namtracename backtraffic\_test.nam ;# назва trace-файлу

# Створення стандартних об’єктів середовища моделювання

set ns [new Simulator]

set tracefd [open backtraffic\_test.tr w]

set namtrace [open $namtracename w]

$ns trace-all $tracefd

$ns namtrace-all-wireless $namtrace $val(x) $val(y)

$ns puts-nam-traceall {# nam4wpan #}

Mac/802\_15\_4 wpanCmd verbose on

Mac/802\_15\_4 wpanNam namStatus on

# Параметри моделі розповсюдження

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) 1.20174e-07

Phy/WirelessPhy set CSThresh\_ $dist(40m)

Phy/WirelessPhy set RXThresh\_ $dist(40m)

set topo [new Topography]

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

create-god $val(nn)

# Завантаження параметрів MobileNode

$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 \

-movementTrace OFF \

-rxPower 35.28e-3 \

-txPower 31.32e-3 \

-idlePower 712e-6 \

-sleepPower 144e-9

# Створення об’єктів MobileNode

for {set i 0} {$i < $val(nn) } { incr i } {

set mnode\_($i) [$ns node]

}

# Функція, що встановлює флаг GTS\_delivery\_ на рівні LL вузла

proc setNodeGTS {node gts} {

set mac [ [set node] getMac 0]

puts $mac

set ll [$mac up-target]

puts $ll

$ll set GTS\_delivery\_ $gts

}

if {[info exists val(GTS\_setting)]} {

# Сервер веде передачу протягом CFP в інтервалах GTS

setNodeGTS $mnode\_(0) 1

}

# Функція, що встановлює розмір черги інтерфейсу окремого об’єкту MobileNode

proc setNodeIfqLen {node qlen} {

set mac [ [set node] getMac 0]

puts $mac

set ll [$mac up-target]

puts $ll

set ifq [$ll down-target]

$ifq set limit\_ $qlen

}

# Розмір черги інтерфейсу Сервера = 1000

setNodeIfqLen $mnode\_(0) 1000

# Створення таблиці маршрутизації для NOAH на Сервері

set cmd "[$mnode\_(0) set ragent\_] routing $val(nn) 0 0"

for {set to 1} {$to < $val(nn) } {incr to} {

set hop $to

set cmd "$cmd $to $hop"

}

eval $cmd

# Створення таблиць маршрутизації для NOAH на Клієнатх

for {set i 1} {$i < $val(nn) } {incr i} {

set cmd "[$mnode\_($i) set ragent\_] routing $val(nn)"

for {set to 0} {$to < $val(nn) } {incr to} {

if {$to == $i} {

set hop $to

} else {

set hop 0

}

set cmd "$cmd $to $hop"

}

eval $cmd

}

# Розташування вузлів Клієнтів випадково на площині

for {set i 1} {$i < $val(nn) } { incr i } {

$mnode\_($i) set X\_ [ expr {$val(x) \* rand()} ]

$mnode\_($i) set Y\_ [ expr {$val(y) \* rand()} ]

$mnode\_($i) set Z\_ 0

}

# Розташування Сервера посередені площини

$mnode\_(0) set X\_ [ expr {$val(x)/2} ]

$mnode\_(0) set Y\_ [ expr {$val(y)/2} ]

$mnode\_(0) set Z\_ 0.0

$mnode\_(0) label "Sink"

for {set i 0} {$i < $val(nn)} { incr i } {

$ns initial\_node\_pos $mnode\_($i) 10

}

#

# Об’єднання вузлів у мережу

#

# Старт координатора

$ns at 0.0 "$mnode\_(0) sscs startPANCoord $val(beacon\_enabled) $val(BO) $val(SO)"

if {[info exists val(GTS\_setting)]} {

# Встановлення параеметрів GTS

$ns at $val(assocStart) "$mnode\_(0) sscs MLME\_GTS\_indication 0 [expr {$val(GTS\_setting)}]"

}

# Підключення Клиєнтів до Сервера

for {set i 1} {$i < $val(nn)} { incr i } {

set t [expr $val(assocStart) + $val(assocTime) \* ($i - 1)]

$ns at $t "$mnode\_($i) sscs startDevice $val(beacon\_enabled) 0 0 $val(BO) $val(SO)"

}

#

# Настройка генераторів трафіку

#

# Створення агентів широкомовного трафіку

for {set i 0} {$i < $val(nn)} { incr i } {

set agent($i) [new Agent/Broadcastbase]

$mnode\_($i) attach $agent($i) 250

$agent($i) set fid\_ $i

set game($i) [new Application/BroadcastbaseApp]

$game($i) set bsize\_ $val(bmsg-size)

$game($i) set bmsg-interval\_ $val(bmsg-interval)

$game($i) set propagate\_ 0

$game($i) attach-agent $agent($i)

}

# Запуск та останов відправки широкомовного трафіку Сервером у запланований час

$ns at $val(bmsg-start) "$game(0) start "

$ns at $val(bmsg-stop) "$game(0) stop "

# Функція настройки пуасоновського генератору трафіку

proc poissontraffic { src dst } {

global ns val mnode\_

set udp($src) [new Agent/UDP]

eval $ns attach-agent \$mnode\_($src) \$udp($src)

set null($dst) [new Agent/Null]

eval $ns attach-agent \$mnode\_($dst) \$null($dst)

set expl($src) [new Application/Traffic/Exponential]

eval \$expl($src) set packetSize\_ \$val(pois-size)

eval \$expl($src) set burst\_time\_ 0

eval \$expl($src) set idle\_time\_ [expr \$val(pois-interval)\*1000.0-\$val(pois-size)\*8/\$val(pois-rate)]ms ;# idle\_time + pkt\_tx\_time = interval

eval \$expl($src) set rate\_ \$val(pois-rate)k

eval \$expl($src) attach-agent \$udp($src)

eval $ns connect \$udp($src) \$null($dst)

$ns at $val(pois-start) "$expl($src) start"

$ns at $val(pois-stop) "$expl($src) stop"

}

# Створення генераторів трафіку від Клієнтів до Сервера

for {set i 1} {$i < $val(nn)} { incr i } {

poissontraffic $i 0

}

# Планування, коли зупинити роботу вузлів

for {set i 0} {$i < $val(nn) } { incr i } {

$ns at $val(stop) "$mnode\_($i) reset;"

}

# Планування останову моделі

$ns at $val(stop) "$ns nam-end-wireless $val(stop)"

$ns at $val(stop) "stop"

$ns at [expr $val(stop) + 0.01] "puts \"end simulation\"; $ns halt"

proc stop {} {

global ns tracefd namtrace

$ns flush-trace

}

# Запуск моделювання

$ns run