Pearls in Life

Technical reflections and records of thoughts

- RSS

Simulate Random MAC Protocol in NS2 (Part I)

Dec 13th, 2013

In this network project, we would need to:

- Write an simulator using TCL
- Add an new MAC protocol to NS2
- Analyze the simulation results

Let's tackle them one by one. In this post, we'll mainly focus on the simulator part.

Get Familiar with TCL

TCL is actually a quite simple language. It's designed for fast scripting and glue things together. You can find many tutorials online. I found this one especially clean, and straightforward.

Simulator Parameters

First, let's define some parameters that we'll use later.

```
1  # ======================================================================
2  # Project parameters
3  # ======================================================================
4  set val(node_num)       101
5  set val(duration)       10
6  set val(packetsize)     16
7  set val(repeatTx)       10
8  set val(interval)       0.02
9  set val(dimx)           50
10 set val(dimy)           50
11 set val(nam_file)       "jinghaos_pa3.nam"
```

```
12 set val(trace_file)     "jinghaos_pa3.tr"
13 set val(stats_file)     "jinghaos_pa3.stats"
14 set val(node_size)      5
15
16 # ======================================================================
17 # Node options
18 # ======================================================================
19 set val(chan)           Channel/WirelessChannel    ;# channel type
20 set val(prop)           Propagation/TwoRayGround   ;# radio-propagation model
21 set val(netif)          Phy/WirelessPhy            ;# network interface type
22 set val(mac)            Mac/RMAC                 ;# MAC type
23 #set val(mac)            Mac/802_11                ;# MAC type
24 set val(ifq)            Queue/DropTail/PriQueue    ;# interface queue type
25 set val(ll)             LL                         ;# link layer type
26 set val(ant)            Antenna/OmniAntenna        ;# antenna model
27 set val(ifqlen)         50                         ;# max packet in ifq
28 set val(nn)             $val(node_num)                        ;# number of mobilenodes
29 set val(rp)             DSDV                       ;# routing protocol
```

The first part is parameters from the project specification. Here we have 101 nodes (100 source node plus 1 sink node), simulation duration, packet rate, terrain size, etc.

The second part is for node configuration. Here we use `WirelessChannel` with `DSDV` routing protocol. Note that for MAC protocol, we use `Mac/RMAC`, which stands for the random MAC protocol we'll add to NS2. Of course, at this point, we don't have our RMAC protocol yet, so you can substitute it with `Mac/802_11` for the moment.

Simulator Configuration

We can obtain an instance of the simulator, and configure it this way.

```
1  # ======================================================================
2  # Global variables
3  # ======================================================================
4  set ns                     [new Simulator]
5  set tracefd                 [open $val(trace_file) w]
6  set nam                     [open $val(nam_file) w]
7  set stats                   [open $val(stats_file) w]
8  $ns namtrace-all-wireless   $nam $val(dimx) $val(dimy)
9  $ns trace-all               $tracefd
10 set topo                    [new Topography]
11 $topo load_flatgrid         $val(dimx) $val(dimy)
```

Here we set up various global variables, including trace and stats file fd, and also the topology.

The we configure the node.

```
1  #
2  # Create God
3  #
4  create-god $val(nn)
5
6  #Mac/RMAC set repeatTx_ $val(repeatTx)
7  #Mac/RMAC set interval_ $val(interval)
8
9  $ns node-config \
10         -adhocRouting $val(rp) \
11         -llType $val(ll) \
12         -macType $val(mac) \
```

```
13         -ifqType $val(ifq) \
14         -ifqLen $val(ifqlen) \
15         -antType $val(ant) \
16         -propType $val(prop) \
17         -phyType $val(netif) \
18         -channelType $val(chan) \
19         -topoInstance $topo \
20         -agentTrace ON \
21         -routerTrace ON \
22         -macTrace ON \
23         -movementTrace OFF
```

Here we first create an General Operations Director(GOD) object to track the nodes' position in the topology grid. Then we configure the nodes using the parameters we set up earlier.

Note that, again at this point we don't have a RMAC protocol, so we can just comment out the two lines that configure RMAC for now.

The Only Sink Node

Next, we're going to create the sink node.

```
1 #
2 # The only sink node
3 #
4 set sink_node [$ns node]
5 $sink_node random-motion 0
6 $sink_node set X_ [expr $val(dimx)/2]
7 $sink_node set Y_ [expr $val(dimy)/2]
8 $sink_node set Z_ 0
9 $ns initial_node_pos $sink_node $val(node_size)
10
11 set sink [new Agent/LossMonitor]
12 $ns attach-agent $sink_node $sink
```

Here we place the sink node at the center of the terrain, and attach an `LossMonitor` to it, so that we can get the packet statistics. Although the project specification requires us to get the packet statistics from the trace file, we can use the results from `LossMonitor` to verify that analysis results.

The Source Nodes

We need to create 100 source nodes, they should scatter the whole terrain randomly, also, they should start transmission also randomly, which has two benefits: – In practice, they're highly unlikable to synchronize perfectly, so we can simulator real world better. – By starting randomly, we're minimizing the chances they have collision.

So we'll have two random number generators, one for the position, and one for the starting time.

```
1 #
2 # Set up random number generator, to scatter the source nodes
3 #
4 set rng [new RNG]
5 $rng seed 0
6
7 set xrand [new RandomVariable/Uniform]
8 $xrand use-rng $rng
9 $xrand set min_ [expr -$val(dimx)/2]
```

```
10 $xrand set max_ [expr $val(dimx)/2]
11
12 set yrand [new RandomVariable/Uniform]
13 $yrand use-rng $rng
14 $yrand set min_ [expr -$val(dimy)/2]
15 $yrand set max_ [expr $val(dimy)/2]
16
17 set trand [new RandomVariable/Uniform]
18 $trand use-rng $rng
19 $trand set min_ 0
20 $trand set max_ $val(interval)
```

Also note that we set the seed to the Random Number Generator (RNG) to a constant value `0`, so that in each simulation we can get the same results, easy for debug and also analyzing.

Then we create all the source nodes in a `for` loop.

```
1  #
2  # Create all the source nodes
3  #
4  for {set i 0} {$i < $val(nn)-1 } {incr i} {
5      set src_node($i) [$ns node]
6      $src_node($i) random-motion 0
7      set x [expr $val(dimx)/2 + [$xrand value]]
8      set y [expr $val(dimx)/2 + [$xrand value]]
9      $src_node($i) set X_ $x
10     $src_node($i) set Y_ $y
11     $src_node($i) set Z_ 0
12     $ns initial_node_pos $src_node($i) $val(node_size)
13
14     set udp($i) [new Agent/UDP]
15     $udp($i) set class_ $i
16     $ns attach-agent $src_node($i) $udp($i)
17     $ns connect $udp($i) $sink
18
19     set cbr($i) [new Application/Traffic/CBR]
20     $cbr($i) set packet_size_ $val(packetsize)
21     $cbr($i) set interval_ $val(interval)
22     $cbr($i) attach-agent $udp($i)
23     set start [$trand value]
24     $ns at $start "$cbr($i) start"
25     $ns at $val(duration) "$cbr($i) stop"
26 }
```

Note that we use UDP here instead of TCP, since we don’t need any reliable transfer or congestion control from up layer. Also, we attach an Constant Bit Generator (CBR) as the application.

Simulator Control

We first define the actions to take when the simulator stops.

```
1  proc stop {} {
2      global ns tracefd nam stats val sink
3
4      set bytes [$sink set bytes_]
5      set losts  [$sink set nlost_]
6      set pkts [$sink set npkts_]
7      puts $stats "bytes losts pkts"
```

```
8       puts $stats "$bytes $losts $pkts"
9
10      $ns flush-trace
11      close $nam
12      close $tracefd
13      close $stats
14  }
```

Here we first get the packet statistics from `LossMonitor`, and write them to the stats file, then we flush ns trace and close all the files.

Finally, we start the simulator.

```
1 puts "Starting Simulation..."
2 $ns run
```

You can find the complete tcl file here.

Change History

- **Sun Dec 15 11:32:17 2013**
  change tags
- **Sun Dec 15 11:22:27 2013**
  new post

View on Github

tcl

Posted by Jinghao Shi Dec 13th, 2013 In Category: network Tagged with: ns2,

Related Posts

- Simulate Random MAC Protocol in NS2 (Part III)
- OS161 Tool Chain Setup
- Simulate Random MAC Protocol in NS2 (Part IV)
- OS161 Swapping
- Simulate Random MAC Protocol in NS2 (Part II)

« How to Apply Downloaded OTA Package Simulate Random MAC Protocol in NS2 (Part II) »

Comments

Did you know...

If tapped, a Chuck Norris roundhouse kick could power the country of Australia for 44 minutes.

Blog Stats

172 hits

Popular Posts

- OS161 TLB Miss and Page Fault
- OS161 Coremap

- OS161 fork System Call
- OS161 File System Calls
- OS161 execv System Call

Recent Posts

- Tap Notification to Send Email
- Sum of N Largest Numbers in Google Spreadsheet
- OS161 Tool Chain Setup
- Simulate Random MAC Protocol in NS2 (Part IV)
- Simulate Random MAC Protocol in NS2 (Part III)

Tags

AlertDialog C DownloadManager MLFQ SO_REUSEADDR addrspace backup beamer bind binutils bison bmake c c++ caption cat chrome close compare_xml coremap cron cscope ctags dispatch django dropbox dup2 email exit fd_set fdtable fedora figure function gcc gdb getifaddrs glic google calendar graphicspath graphicx grep heap intent jekyll lfs libpopller linux lock lseek mac migration notification ns2 open ota page fault page table pdflatex pid plugin popular posts prototype python query read readline rsync ruby rwlock sbrk scheduling scp sed select service signapk socket sort spreadsheet ssh stack swap synchronization sys161 syscall table tcl texinfo tlb toolchain undodir vimrc vm waitpid wordpress write yacc yum sty zip

Categories

Android (6)
errors (9)
latex (8)
linux (7)
network (8)
octopress (5)
os161 (25)
tricks (1)
vim (4)

GitHub Repos

- latex.template.slides

  Beamer template

- dotfiles

  Various configuration files

- jhshi.github.com

  Personal Blog

- latex.template.acm-proc

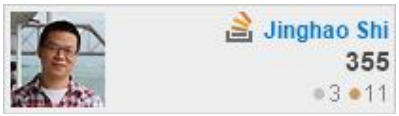
  Latex template for course project proposals, reports.

- latex.template.homework

  Latex template for homework.

@jhshi on GitHub

Stack Overflow



Page Link