

EEC 687/787 Mobile Computing (Spring, 2008)

Ns-2 Laboratory

Chansu Yu

Cleveland State University

Contents

- Download & Install
 - Basic TCL
 - Introduction to ns-2
 - ns-2 overview
 - What can ns-2 do?
 - ns-2 structure
 - How to use ns-2?
 - Simulation in MANETs
 - What is MANETs?
 - Simulation scenario configuration
 - Simulation results analysis
-

Download and Install

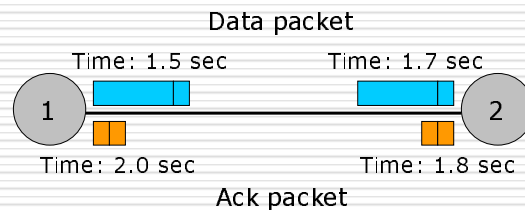
- Reading
 - Wireless and Mobility Extensions to ns-2 (<http://www.isi.edu/nsnam/ns/tutorial/nsindex.html>).
 - Ns2 manual, "Mobile networking in ns," Ch. 16
- Download the latest ns-2 (version 2.30) with additional components source from <http://www.isi.edu/nsnam/dist/ns-allinone-2.30.tar.gz> and put it to your desirable folder, say /home/student.
- Run the following commands at /home/student :
 - % gunzip ns-allinone-2.30.tar.gz
 - % tar -xvf ns-allinone-2.30.tar
- Run the script at /home/student/ns-allinone-2.30 :
" % ./install"
 - This installation script will check your Linux environment, compile and install your ns-2 system.

ns-2 Overview

- What is ns-2?
 - Abbreviation of Network Simulator
 - Discrete event simulator targeted at networking (wired and wireless) research
 - Basically, a TCL interpreter
- Where to get?
 - Free and open source
 - ns website <http://www.isi.edu/nsnam/ns/>
- Working platforms
 - Most UNIX or UNIX-like systems; e.g. Linux
 - Windows (using cygwin)

ns-2: Discrete Event Simulator

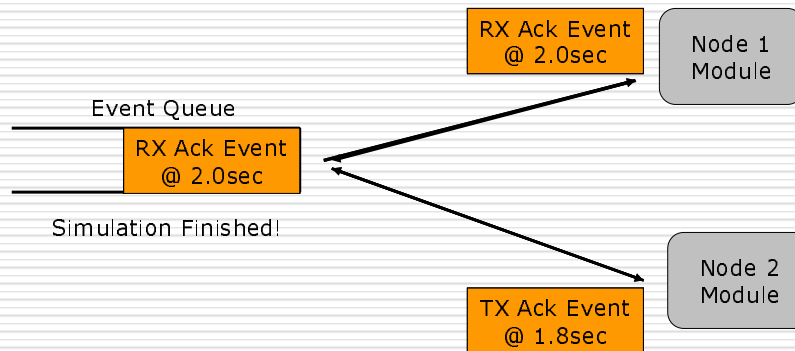
- ns-2 is an discrete event driven simulation
 - Physical activities are translated to events
 - Events are queued and processed in the order of their scheduled occurrences
 - Time progresses as the events are processed



ns-2: Discrete Event Simulator

- $T=1.5$, node 1 has a pkt to tx to node 2 (packet generation event)
 - $T=1.7$, node 2 will receive (packet reception event)
- $T=1.8$, node 2 replies an ack to node 1 (packet generation event)
 - $T=2.0$, node 1 will receive (packet reception event)

Event Driven Simulation



ns-2: Discrete Event Simulator

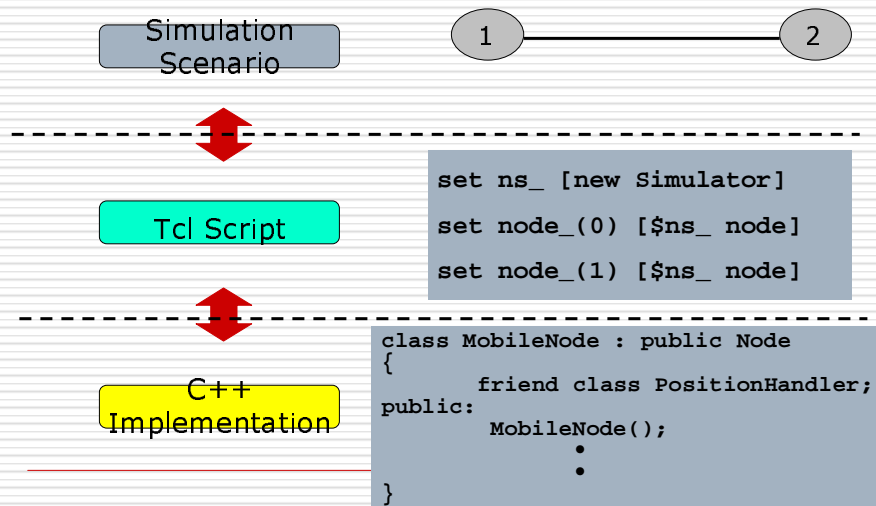
- T=1.5, node 1 has a pkt to tx to node 3
(packet generation event)
 - T=1.7, node 3 will receive
(packet reception event)

- T=1.6, node 2 has a pkt to tx to node 4
(packet generation event)
 - T=1.65, node 4 will receive
(packet reception event)

ns-2 uses two languages? (Tcl & C++)

- C++: Detailed protocol simulations require systems programming language
 - byte manipulation, packet processing, algorithm implementation
 - Run time speed is important
 - Turn around time (run simulation, find bug, fix bug, recompile, re-run) is slower
- Tcl: Simulation of slightly varying parameters or configurations
 - quickly exploring a number of scenarios
 - iteration time (change the model and re-run) is more important

ns-2 Environment



Linux

- We'll use Linux environment.
 - For convenience, we'll use "cygwin" which is the Linux interface on top of Windows.

 - Linux commands???
 - Editors???
-

Basic TCL

(<http://tmml.sourceforge.net/doc/tcl/index.html>)

```
set a 43
set b 27
set c [expr $a + $b]
set d [expr [expr $a - $b] * $c]
for {set k 0} {$k < 10} {incr k} {
    if {$k < 5} {
        puts "k < 5, pow= [expr pow($d, $k)]"
    } else {
        puts "k >= 5, mod= [expr $d % $k]"
    }
}
```

Hello World - Interactive Mode

```
% ns
% set ns [new Simulator]
_o3
% $ns at 1 "puts \"Hello World!\""
1
% $ns at 1.5 "exit"
2
% $ns run
Hello World!
%
```

Hello World - Passive Mode

```
simple.tcl
  set ns [new Simulator]
  $ns at 1 "puts \"Hello World!\""
  $ns at 1.5 "exit"
  $ns run

% ns simple.tcl
Hello World!

%
```

Simulation with ns-2

- Creating the event scheduler

 - Creating network: nodes, links & queue
 - Computing routes
 - Creating connection
 - Creating traffic

 - Inserting errors
 - Tracing

 - Wireless Support
-

Protocols or Controls Implemented in ns2

- Transport layer (traffic agent)
TCP; UDP
 - Network layer (routing agent)
 - Wired
Distance vector; Link state (patch needed)
 - Wireless
AODV; DSR; DSDV; TORA
 - Interface queue
FIFO queue; DropTail queue; Priority queue; etc.
 - Logic link control layer
IEEE 802.2; ARP
-

Protocols or Controls Implemented in ns2 (cont.)

- MAC layer
 - Wired
 - IEEE 802.3 (CSMA/CD)
 - Wireless
 - IEEE 802.11 (CSMA/CA)
 - DCF
 - PCF (partially implemented)
 - Physical layer
 - Wired
 - IEEE 802.3
 - Wireless
 - IEEE 802.11
 - DSSS (Direct Sequence Spread Spectrum)
 - FHSS (Frequency-Hopping Spread Spectrum); not implemented
 - IR (Infrared); not implemented
-

Protocols or Controls Implemented in ns2 (cont.)

- Wireless channel
 - Friss-space model
 - Two-ray ground model
 - Shadowing model
 - Fading model (patch needed)
 - Omni directional antenna
-

How to Use ns-2?

- Design simulation
 - Determine simulation scenario, parameters.
 - Build ns-2 script using tcl
 - If necessary implement algorithm using C++.
 - Run simulation
 - For convenience use shell batch file.
 - Analyze simulation results
 - Use shell command or programming languages.
-

Visualization is Possible in ns-2

- Example
 - nam-1.12/edu/C2-sliding-color.nam
 - nam-1.12/tcl/test-wireless-2.nam
-