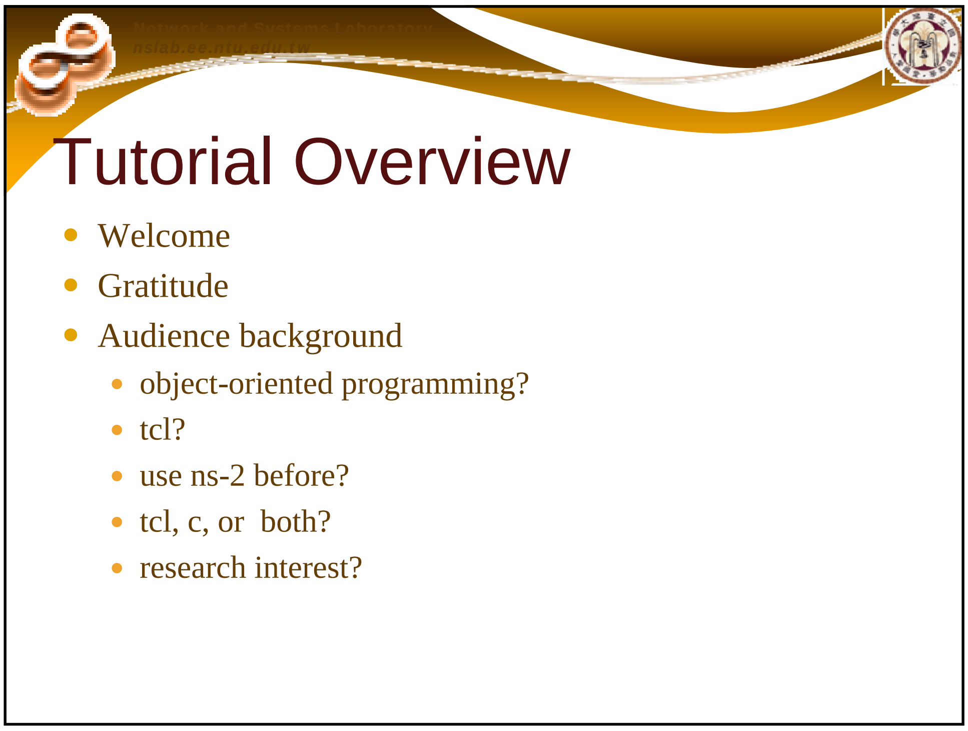
Network Simulation and Testing

Polly Huang
Department of Electrical Engineering
National Taiwan University
http://cc.ee.ntu.edu.tw/~phuang
phuang@cc.ee.ntu.edu.tw

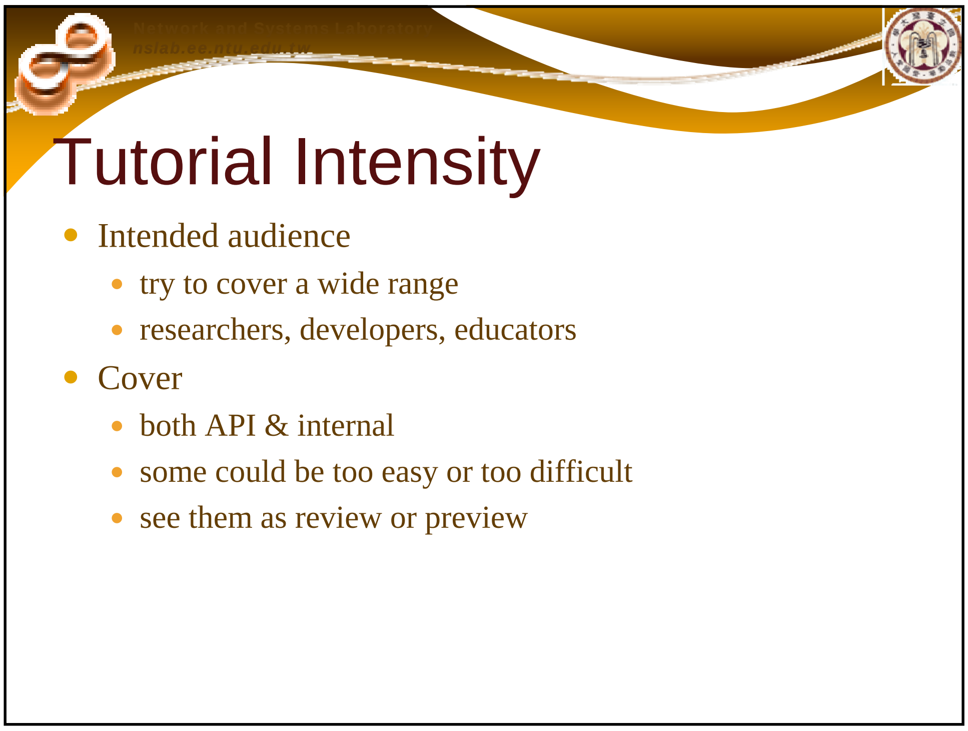
ns-2 Tutorial

In 4 sessions



Tutorial Overview

- Welcome
- Gratitude
- Audience background
  - object-oriented programming?
  - tcl?
  - use ns-2 before?
  - tcl, c, or both?
  - research interest?



Tutorial Intensity

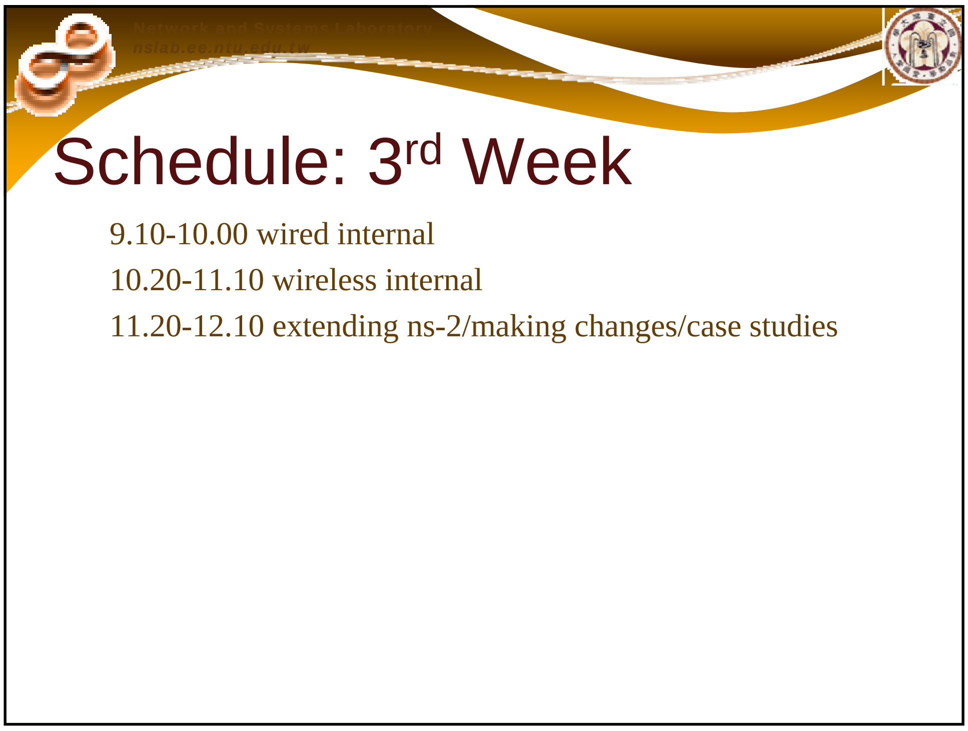
- Intended audience
  - try to cover a wide range
  - researchers, developers, educators
- Cover
  - both API & internal
  - some could be too easy or too difficult
  - see them as review or preview

Schedule: 1st Week

9.10-10.00 overview/intro/essential/getting started
10.20-11.10 tcl/otcl/ns-2 fundamentals
11.20-12.10 examples - TCP, RED, multicast, web, wireless

Schedule: 2nd Week

9.10-10.00 lab 1 setup/running examples
10.20-11.10 lab 2 tcl/otcl exercises
11.20-12.10 lab 3 simple ns-2 exercise

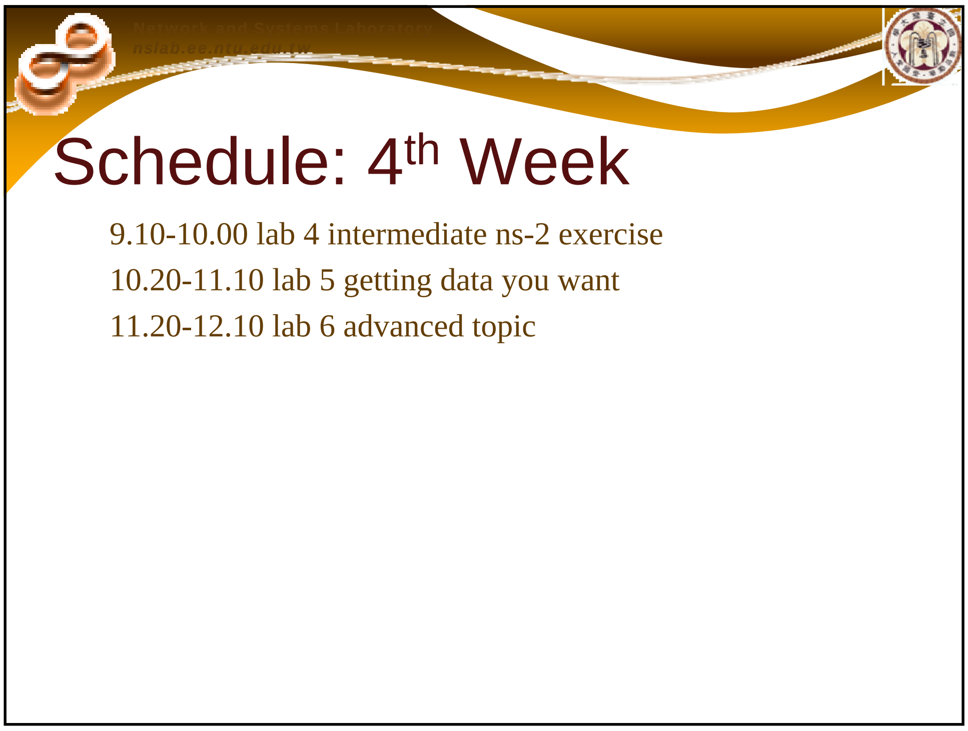


Schedule: 3rd Week

9.10-10.00 wired internal

10.20-11.10 wireless internal

11.20-12.10 extending ns-2/making changes/case studies

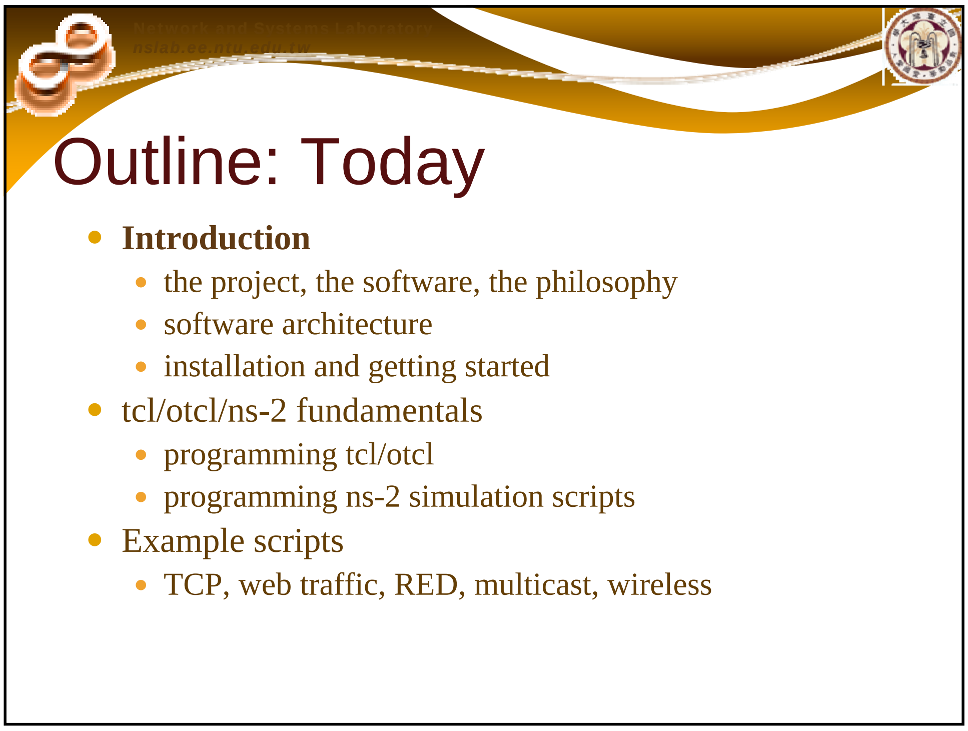


Schedule: 4th Week

9.10-10.00 lab 4 intermediate ns-2 exercise

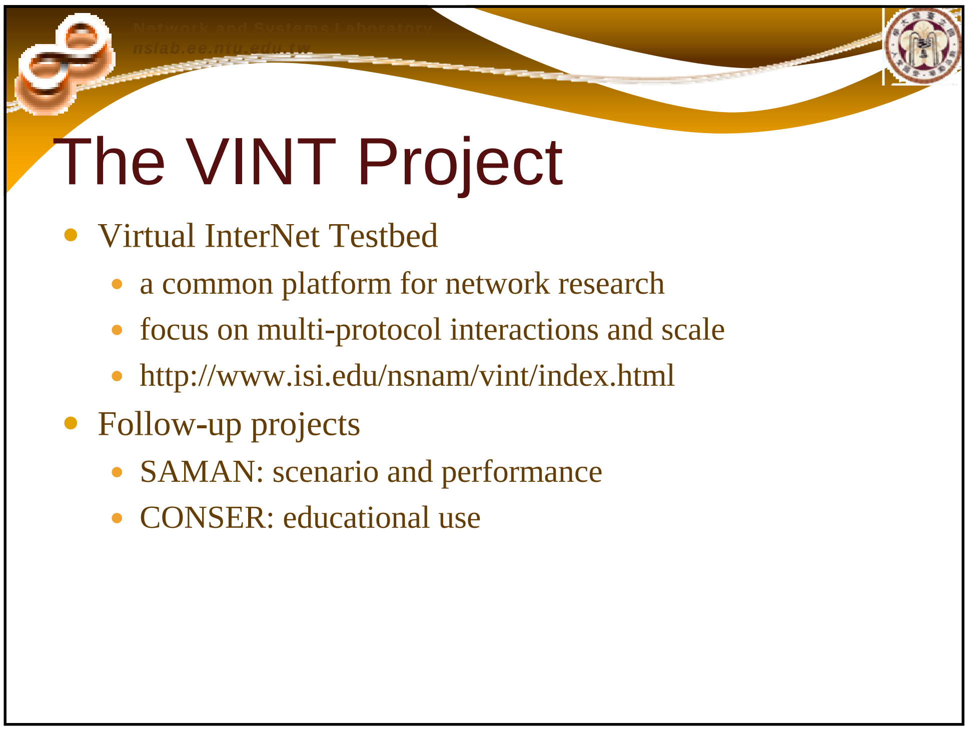
10.20-11.10 lab 5 getting data you want

11.20-12.10 lab 6 advanced topic



Outline: Today

- **Introduction**
  - the project, the software, the philosophy
  - software architecture
  - installation and getting started
- tcl/otcl/ns-2 fundamentals
  - programming tcl/otcl
  - programming ns-2 simulation scripts
- Example scripts
  - TCP, web traffic, RED, multicast, wireless



The VINT Project

- Virtual InterNet Testbed
  - a common platform for network research
  - focus on multi-protocol interactions and scale
  - http://www.isi.edu/nsnam/vint/index.html
- Follow-up projects
  - SAMAN: scenario and performance
  - CONSER: educational use

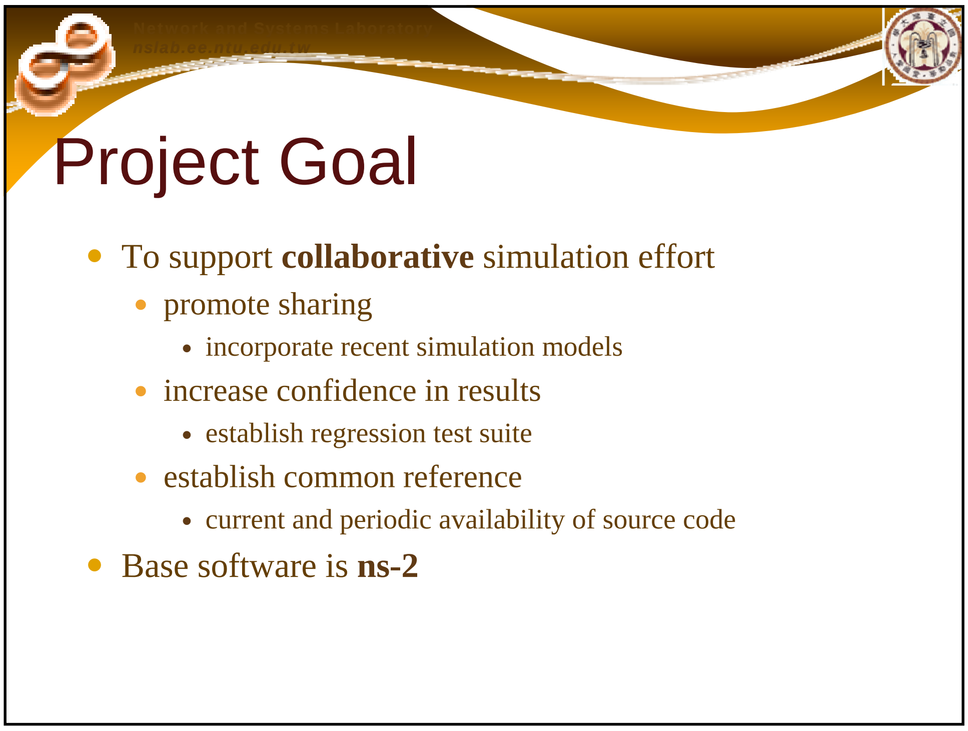
Multi-state collaboration

- USC/ISI
- ACIRI

- UC Berkeley
- CMU
- NTU nsLab
- etc…

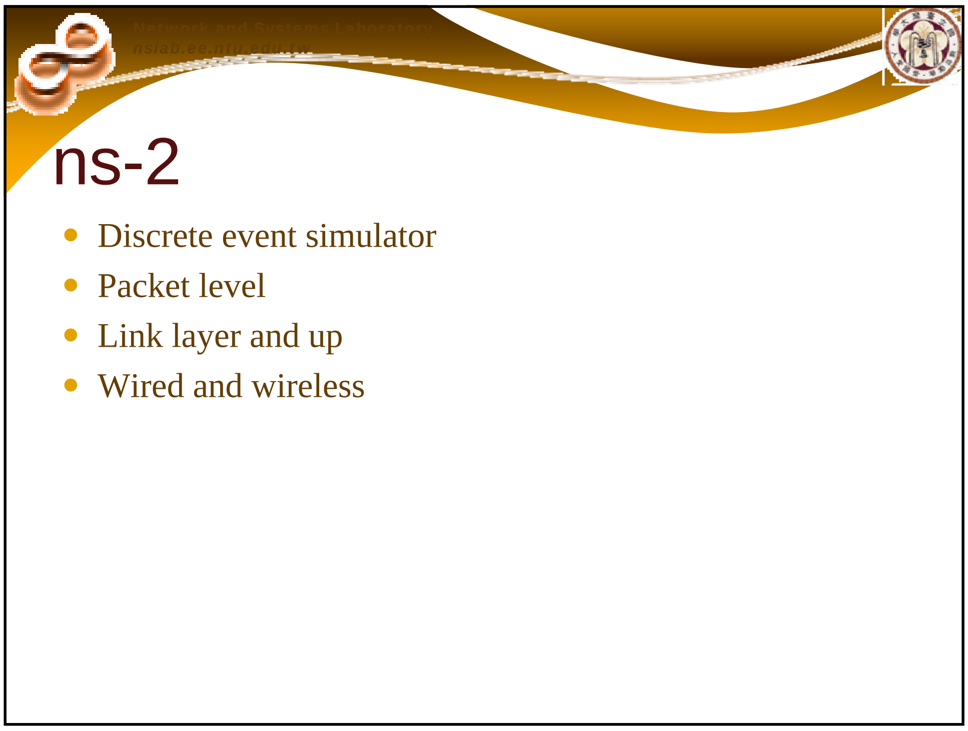
VINT People

- Project leads and co-PIs
  - Lee Breslau (AT&T Labs-Research)
  - Deborah Estrin (UCLA)
  - Kevin Fall (INTEL Research)
  - Sally Floyd (AT&T/ACIRI)
  - Mark Handley (AT&T/ACIRI)
  - John Heidemann (USC/ISI)
  - Scott Shenker (AT&T/ACIRI)
- Graduate students and staff members



Project Goal

- To support **collaborative** simulation effort
  - promote sharing
    - incorporate recent simulation models
  - increase confidence in results
    - establish regression test suite
  - establish common reference
    - current and periodic availability of source code
- Base software is **ns-2**



ns-2

- Discrete event simulator
- Packet level
- Link layer and up
- Wired and wireless

Development Status

- Columbia NEST
- UCB REAL
- ns-1
- ns-2 (as of 2001…)
  - 100K lines of C++ code
  - 70K lines of otcl support code
  - 30K lines of test suites
  - 20K lines of documentation

Usage and Releases

- Users from approximately
  - 600+ institutes
  - 50+ countries
- Releases
  - periodic official releases
  - nightly snapshots (probably compiles and works, but buyers beware)
  - available from USC/ISI or UK mirror

Platforms

- Most UNIX and UNIX-like systems
  - √ FreeBSD or *BSD
  - √ Linux
  - √ Sun Solaris
  - ? HP, SGI
- Window 95/98/NT...
- Emulation only for FreeBSD for now

First Words of Caution

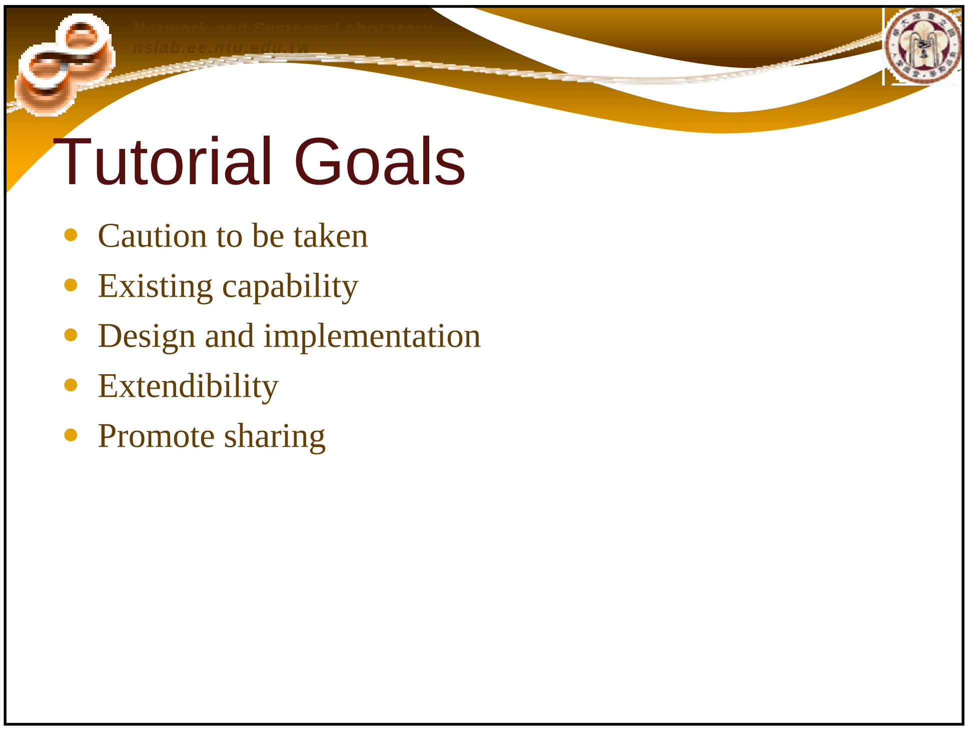
- While we have considerable confidence in ns, ns is **not a polished** and finished product, but the result of an ongoing effort of research and development. In particular, bugs in the software are still being discovered and corrected.

Second Words of Caution

- Users of ns are responsible for verifying for themselves that their simulations are not invalidated by **bugs**. We are working to help the users with this by significantly expanding and automating the validation tests and demos.

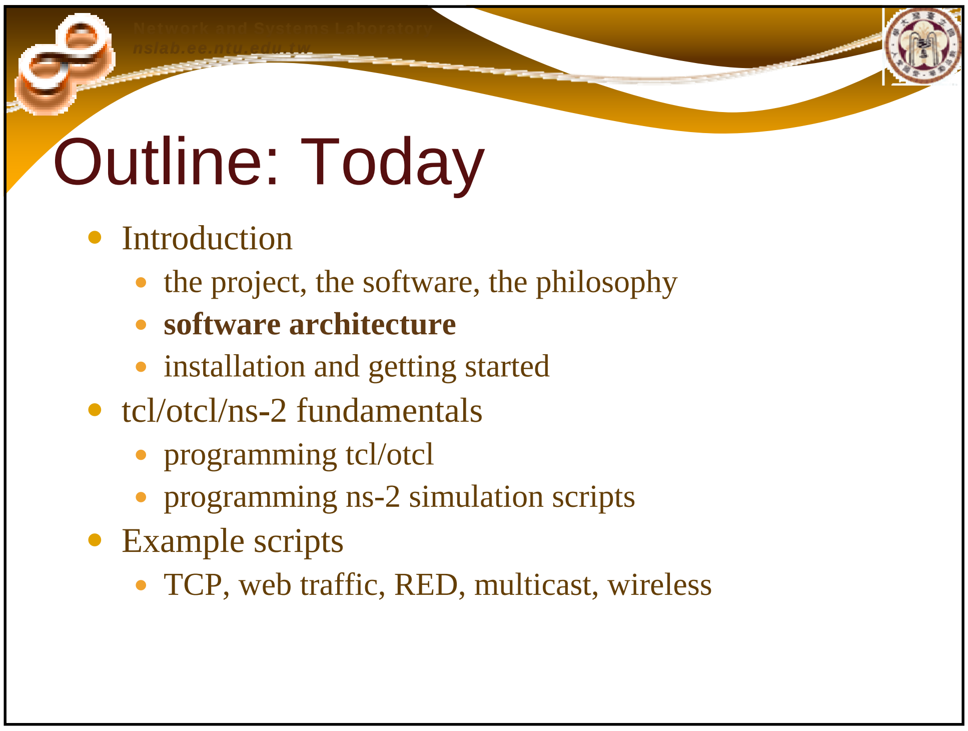
Third Words of Caution

- Similarly, users are responsible for verifying for themselves that their simulations are not invalidated because the **model** implemented in the simulator is not the model that they were expecting. The ongoing ns Notes and Documentation should help in this process.



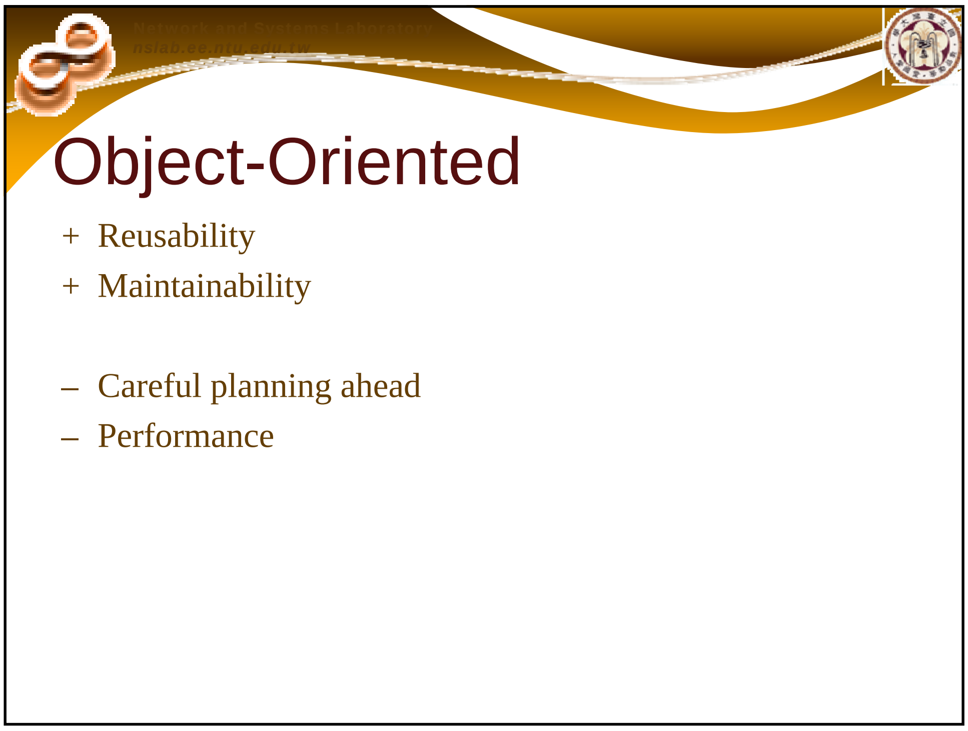
Tutorial Goals

- Caution to be taken
- Existing capability
- Design and implementation
- Extendibility
- Promote sharing



Outline: Today

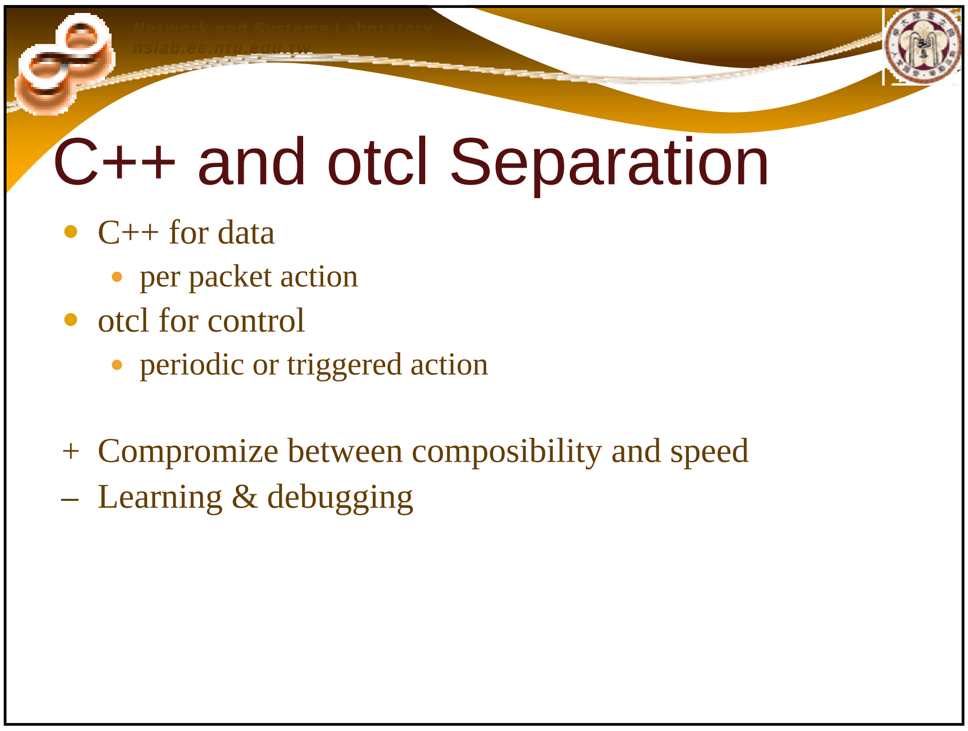
- Introduction
  - the project, the software, the philosophy
  - **software architecture**
  - installation and getting started
- tcl/otcl/ns-2 fundamentals
  - programming tcl/otcl
  - programming ns-2 simulation scripts
- Example scripts
  - TCP, web traffic, RED, multicast, wireless



Object-Oriented

- + Reusability
- + Maintainability

- – Careful planning ahead
- – Performance

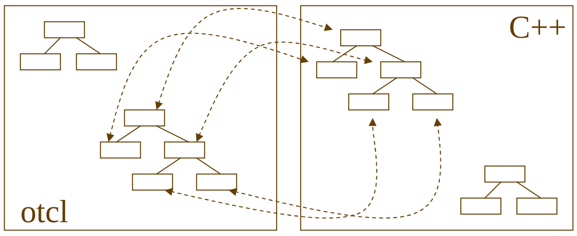


C++ and otcl Separation

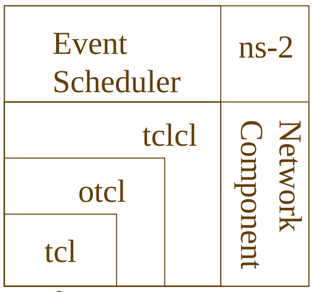
- C++ for data
  - per packet action
- otcl for control
  - periodic or triggered action

- + Compromize between composibility and speed
- – Learning & debugging

otcl and C++: The Duality



tcl Interpreter With Extents



- otcl: object-oriented support
- tclcl: C++ and otcl linkage
- Discrete event scheduler
- Data network components

Installation

- Getting the pieces
  - (tcl/tk8.4.5), otcl, tclcl, ns-2, (and nam-1)
- http://www.isi.edu/nsnam/ns/
- ns-users@isi.edu
  - ns-users-request@isi.edu
  - ‘subscribe ns-users’ in body
- ns-announce@isi.edu

Hello World – Interactive Mode

```
swallow 71% ns
% set ns [new Simulator]
_o3
% $ns at 1 “puts \“Hello World!\””
1
% $ns at 1.5 “exit”
2
% $ns run
Hello World!
swallow 72%
```

Hello World - Passive Mode

simple.tcl

```
set ns [new Simulator]
$ns at 1 “puts \“Hello World!\””
$ns at 1.5 “exit”
$ns run
```

```
swallow 74% ns simple.tcl
Hello World!
swallow 75%
```

Outline: Today

- Introduction
  - the project, the software, the philosophy
  - software architecture
  - installation and getting started
- **tcl/otcl/ns-2 fundamentals**
  - programming tcl/otcl
  - programming ns-2 simulation scripts
- Example scripts
  - TCP, web traffic, RED, multicast, wireless

Fundamentals

- tcl
- otcl
  - ftp://ftp.tns.lcs.mit.edu/pub/otcl/doc/tutorial.html
- ns-2
  - http://www.isi.edu/nsnam/ns/ns_doc.ps.gz
  - http://www.isi.edu/nsnam/ns/ns_doc.pdf
  - http://www.isi.edu/nsnam/ns/doc/index.html

Basic tcl

```
proc test {} {
    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]”
        }
    }
}
test
```

Basic otcl

```
Class mom
mom instproc init {age} {
    $self instvar age_
    set age_ $age
}

mom instproc greet {} {
    $self instvar age_
    puts "$age_ years old mom:
    How are you doing?"
}
```

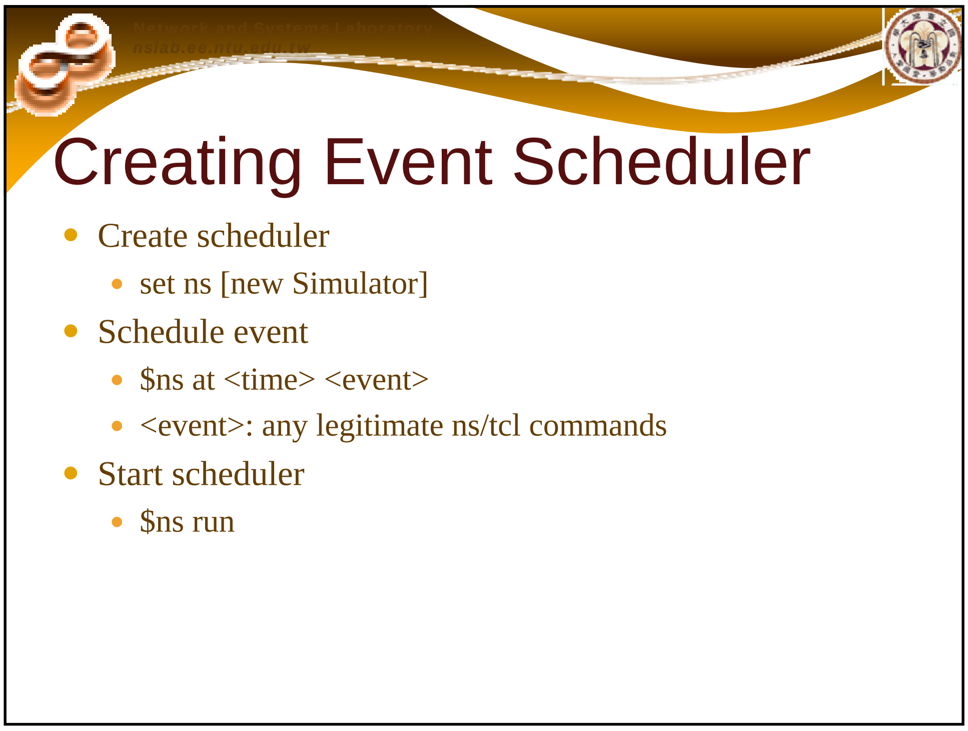
```
Class kid -superclass mom
kid instproc greet {} {
    $self instvar age_
    puts "$age_ years old kid:
    What's up, dude?"
}

set a [new mom 45]
set b [new kid 15]

$a greet
$b greet
```

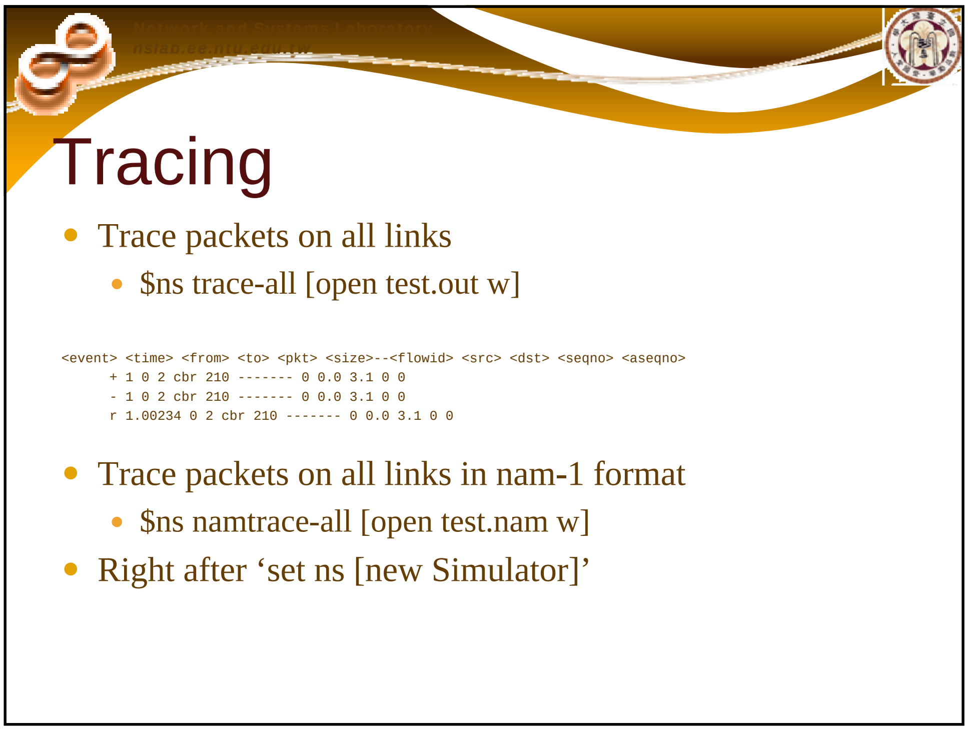
Basic ns-2

- Creating the event scheduler
- [Tracing]
- Creating network
- Computing routes
- Creating connection
- Creating traffic
- Inserting errors



Creating Event Scheduler

- Create scheduler
  - set ns [new Simulator]
- Schedule event
  - $ns at <time> <event>
  - <event>: any legitimate ns/tcl commands
- Start scheduler
  - $ns run

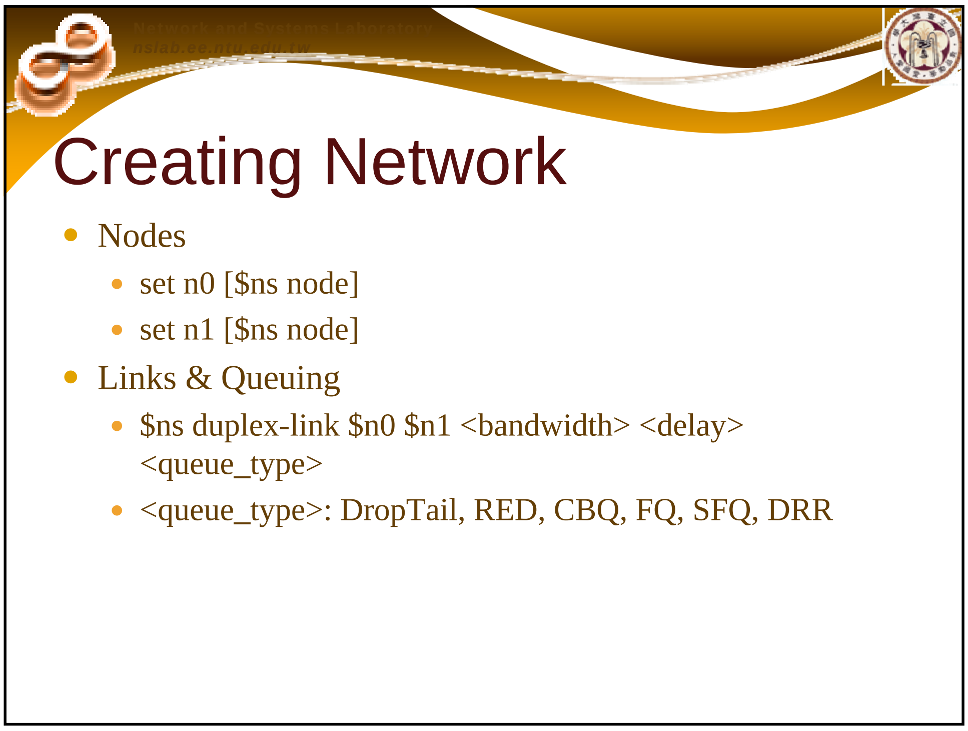


Tracing

- Trace packets on all links
  - $ns trace-all [open test.out w]

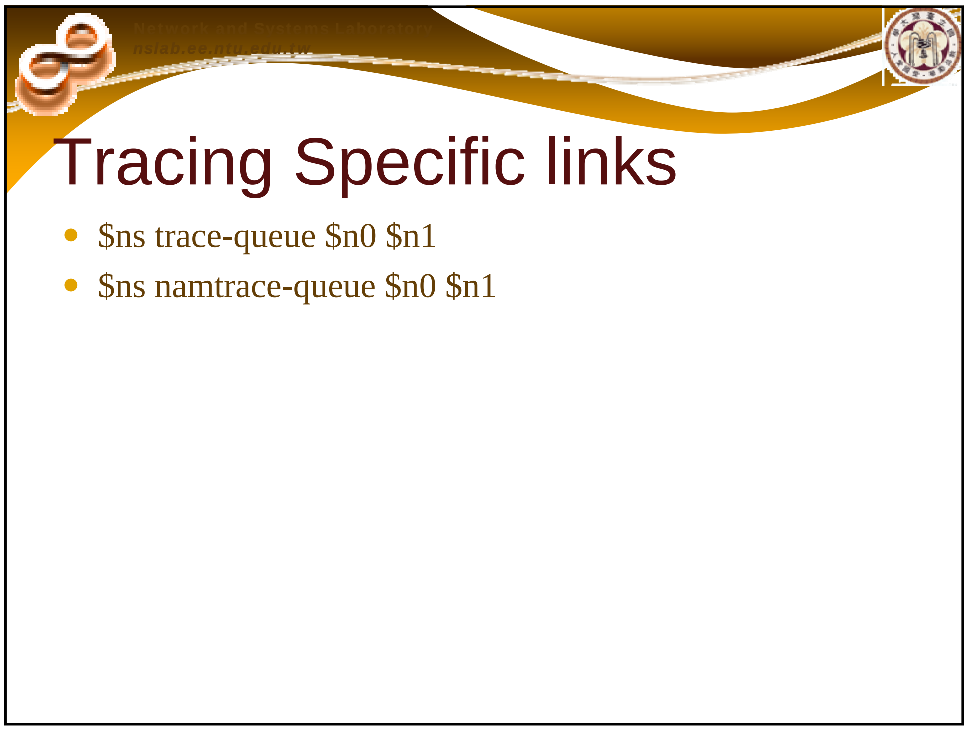
```
<event> <time> <from> <to> <pkt> <size>--<flowid> <src> <dst> <seqno> <aseqno>
      + 1 0 2 cbr 210 ------- 0 0.0 3.1 0 0
      - 1 0 2 cbr 210 ------- 0 0.0 3.1 0 0
      r 1.00234 0 2 cbr 210 ------- 0 0.0 3.1 0 0
```

- Trace packets on all links in nam-1 format
  - $ns namtrace-all [open test.nam w]
- Right after ‘set ns [new Simulator]’



Creating Network

- Nodes
  - set n0 [$ns node]
  - set n1 [$ns node]
- Links & Queuing
  - $ns duplex-link $n0 $n1 <bandwidth> <delay> <queue_type>
  - <queue_type>: DropTail, RED, CBQ, FQ, SFQ, DRR



Tracing Specific links

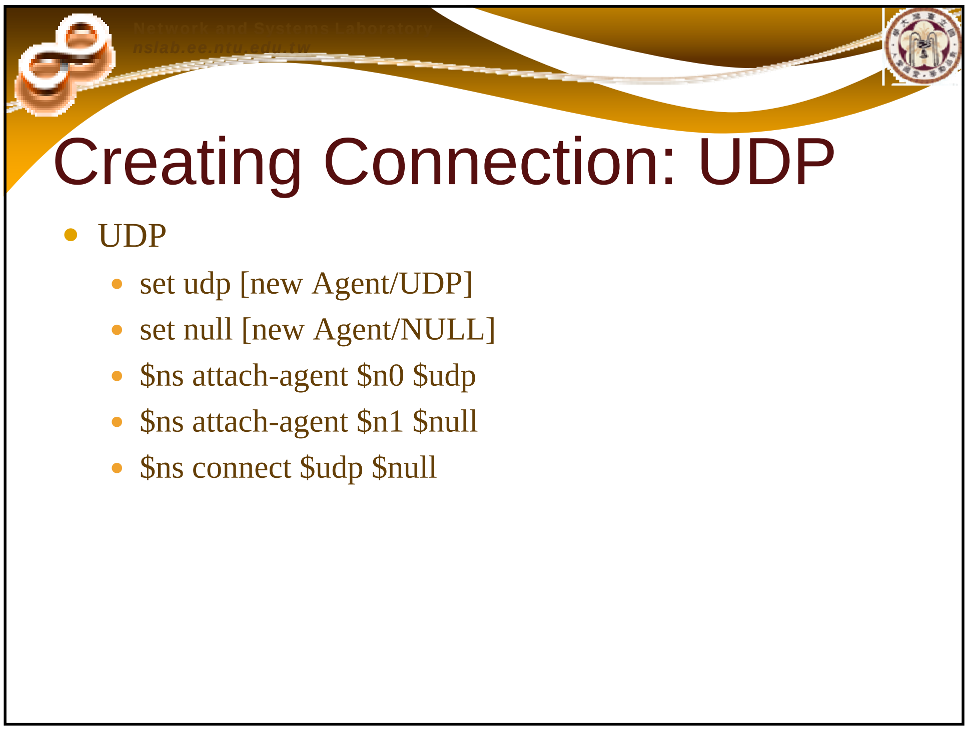
- $ns trace-queue $n0 $n1
- $ns namtrace-queue $n0 $n1

Creating Network: LAN

- LAN
  - $ns make-lan <node_list> <bandwidth> <delay> <ll_type> <ifq_type> <mac_type> <channel_type>
  - <ll_type>: LL
  - <ifq_type>: Queue/DropTail,
  - <mac_type>: MAC/802_3
  - <channel_type>: Channel

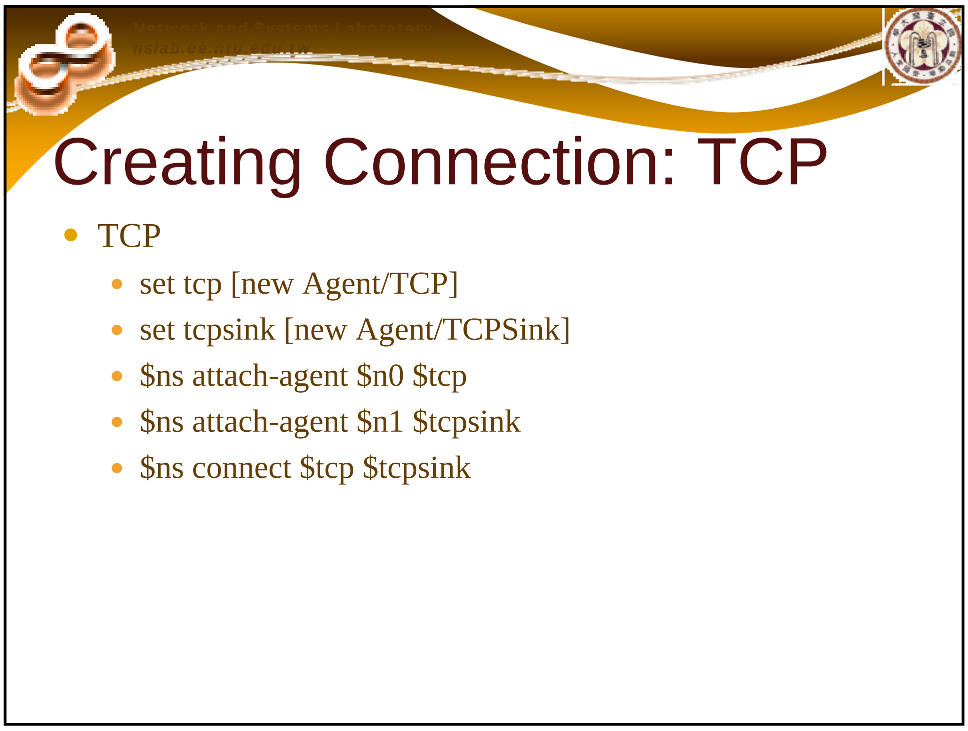
Computing routes

- Unicast
  - $ns rtproto <type>
  - <type>: Static, Session, DV, cost, multi-path



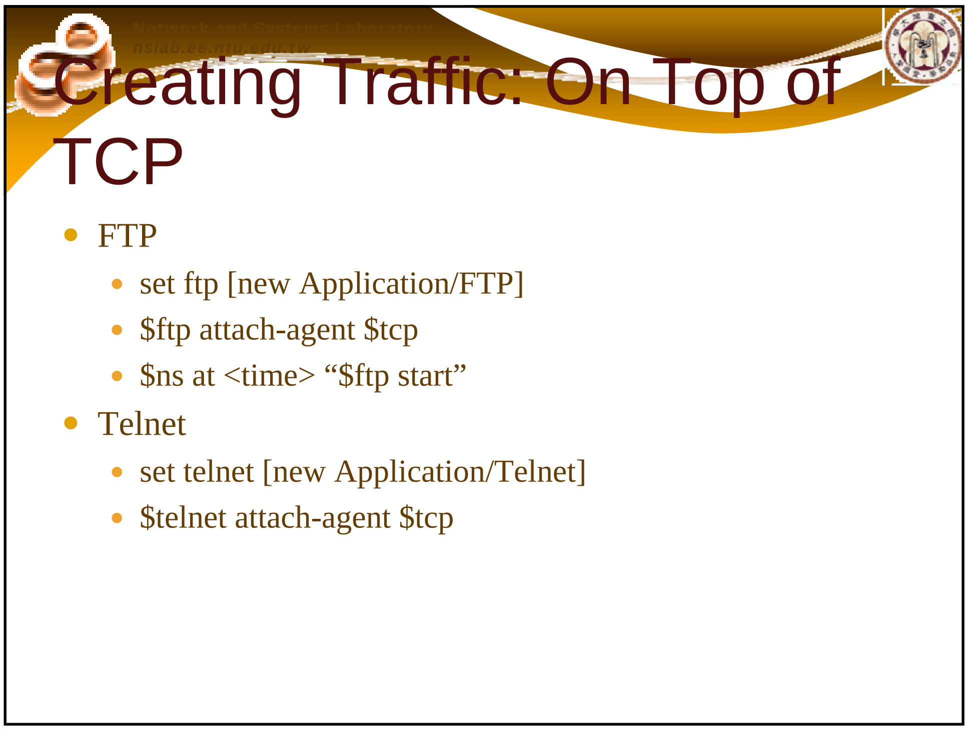
Creating Connection: UDP

- UDP
  - set udp [new Agent/UDP]
  - set null [new Agent/NULL]
  - $ns attach-agent $n0 $udp
  - $ns attach-agent $n1 $null
  - $ns connect $udp $null



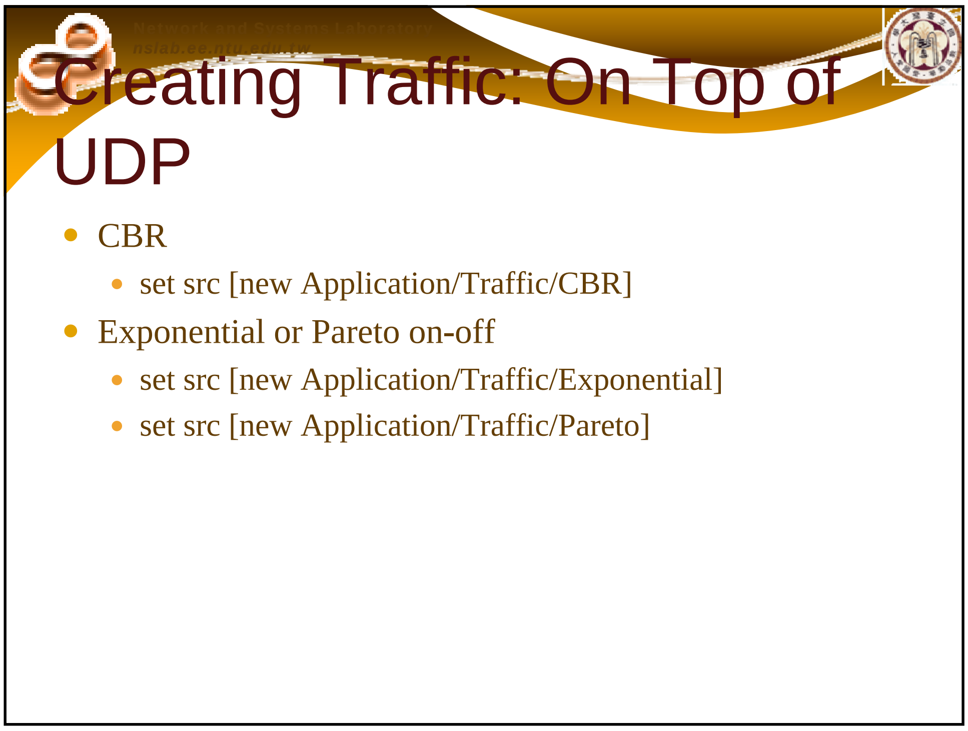
Creating Connection: TCP

- TCP
  - set tcp [new Agent/TCP]
  - set tcpsink [new Agent/TCPSink]
  - $ns attach-agent $n0 $tcp
  - $ns attach-agent $n1 $tcpsink
  - $ns connect $tcp $tcpsink



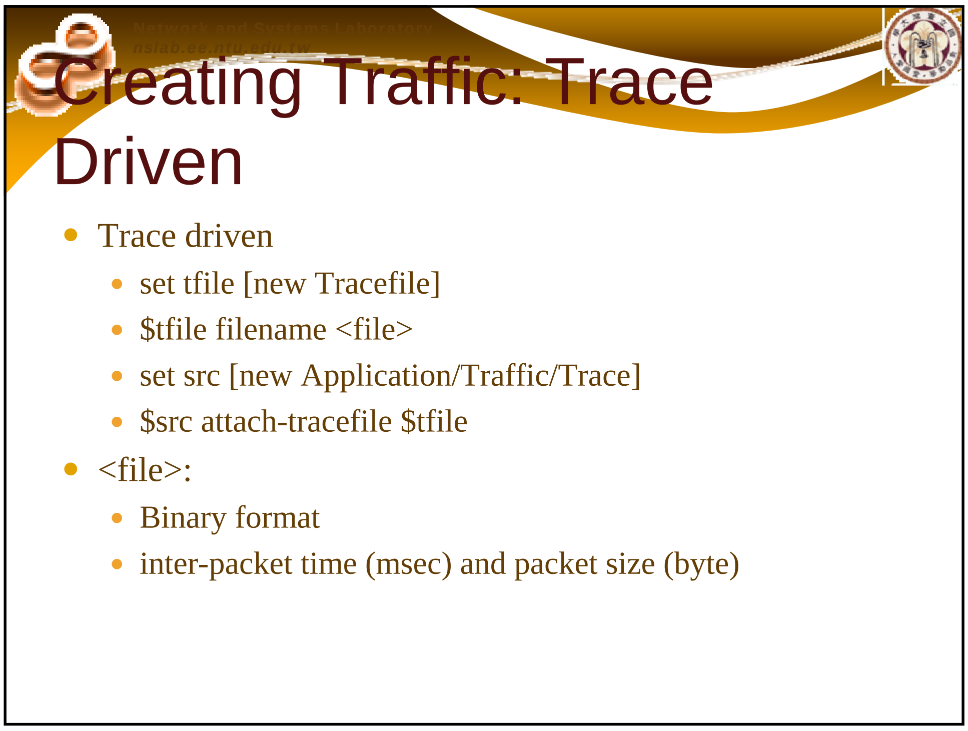
Creating Traffic: On Top of TCP

- FTP
  - set ftp [new Application/FTP]
  - $ftp attach-agent $tcp
  - $ns at <time> "$ftp start"
- Telnet
  - set telnet [new Application/Telnet]
  - $telnet attach-agent $tcp



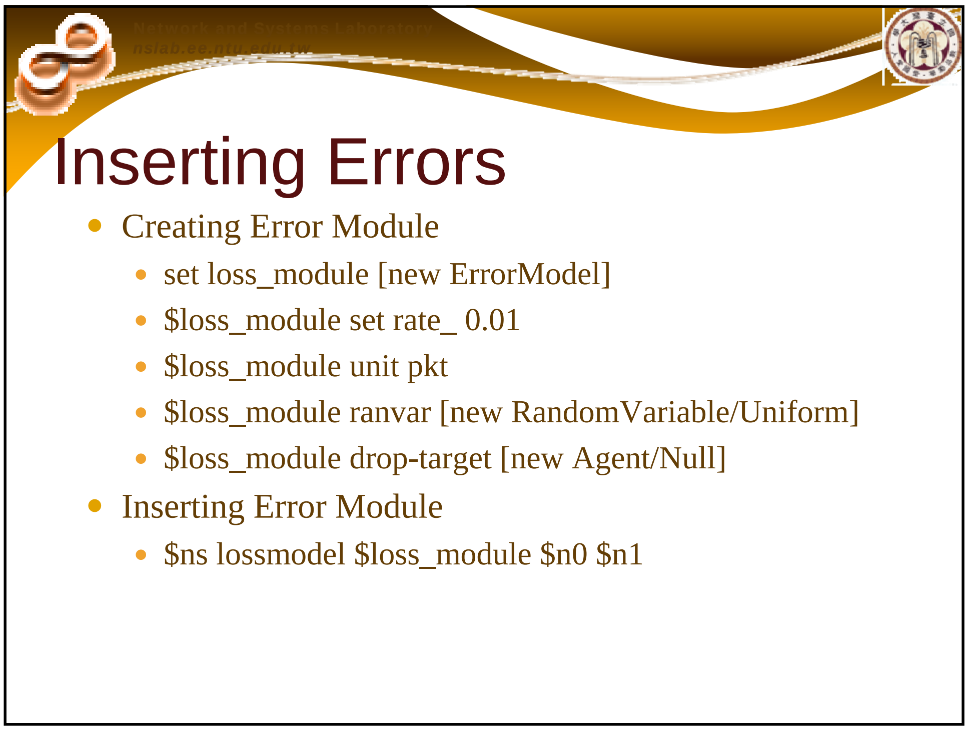
Creating Traffic: On Top of UDP

- CBR
  - set src [new Application/Traffic/CBR]
- Exponential or Pareto on-off
  - set src [new Application/Traffic/Exponential]
  - set src [new Application/Traffic/Pareto]



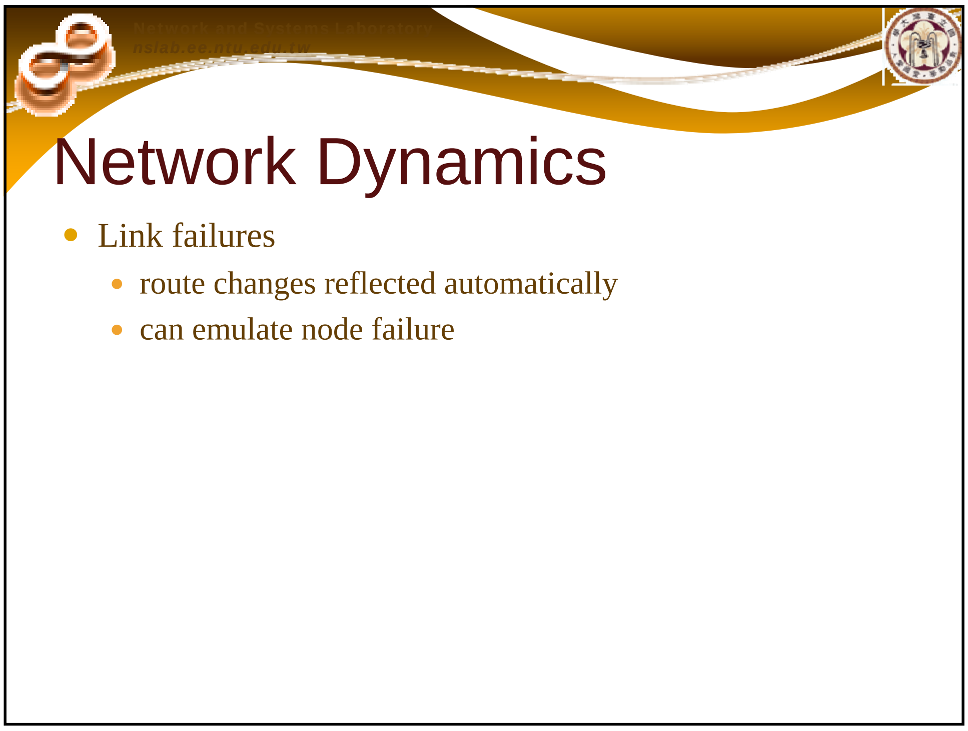
Creating Traffic: Trace Driven

- Trace driven
  - set tfile [new Tracefile]
  - $tfile filename <file>
  - set src [new Application/Traffic/Trace]
  - $src attach-tracefile $tfile
- <file>:
  - Binary format
  - inter-packet time (msec) and packet size (byte)



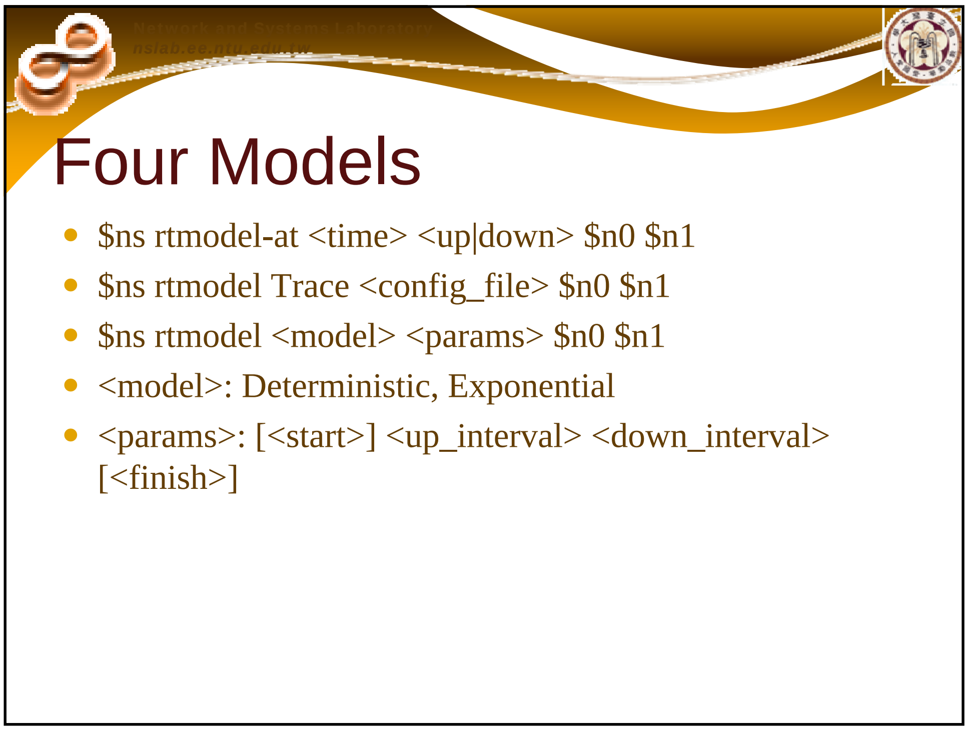
Inserting Errors

- Creating Error Module
  - set loss_module [new ErrorModel]
  - $loss_module set rate_ 0.01
  - $loss_module unit pkt
  - $loss_module ranvar [new RandomVariable/Uniform]
  - $loss_module drop-target [new Agent/Null]
- Inserting Error Module
  - $ns lossmodel $loss_module $n0 $n1



Network Dynamics

- Link failures
  - route changes reflected automatically
  - can emulate node failure



Four Models

- $ns rtmodel-at <time> <up|down> $n0 $n1
- $ns rtmodel Trace <config_file> $n0 $n1
- $ns rtmodel <model> <params> $n0 $n1
- <model>: Deterministic, Exponential
- <params>: [<start>] <up_interval> <down_interval> [<finish>]

Outlines

- Essentials
- Getting Started
- Fundamental tcl, otcl and ns-2
- **Case Studies - TCP, web traffic, RED**

Case Studies

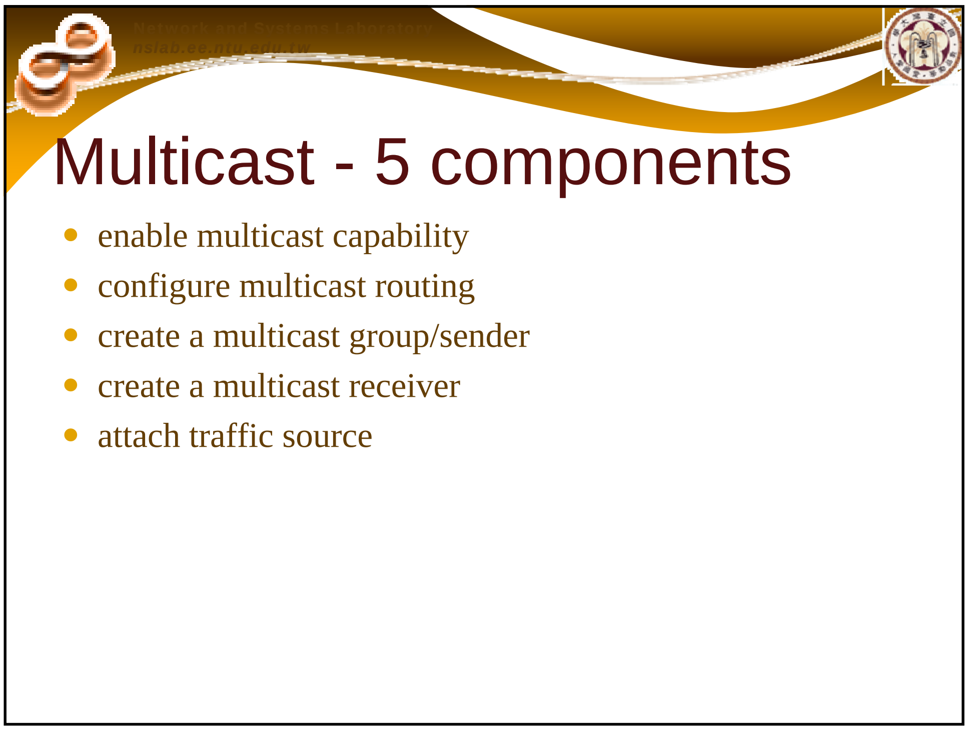
- TCP (tcp.tcl)
- Web (web.tcl & dumbbell.tcl)
- Queuing - RED (red.tcl)

Visualization Tools

- nam-1 (Network AniMator Version 1)
- xgraph

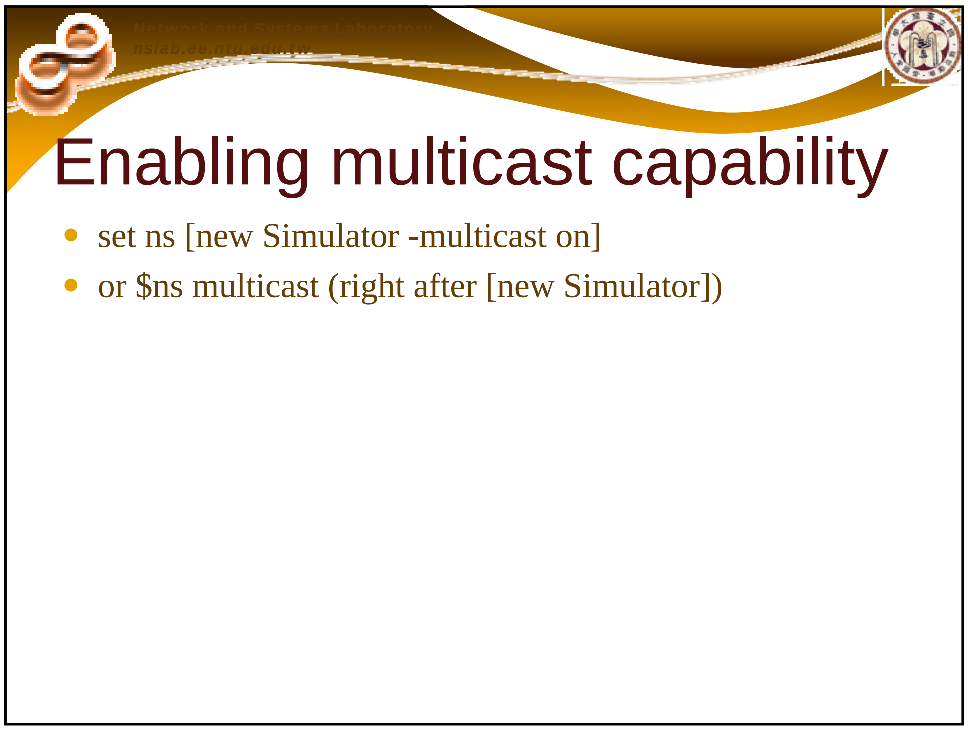
Basic ns-2: Special Topics

- multicast support
- application-level support
- wireless support



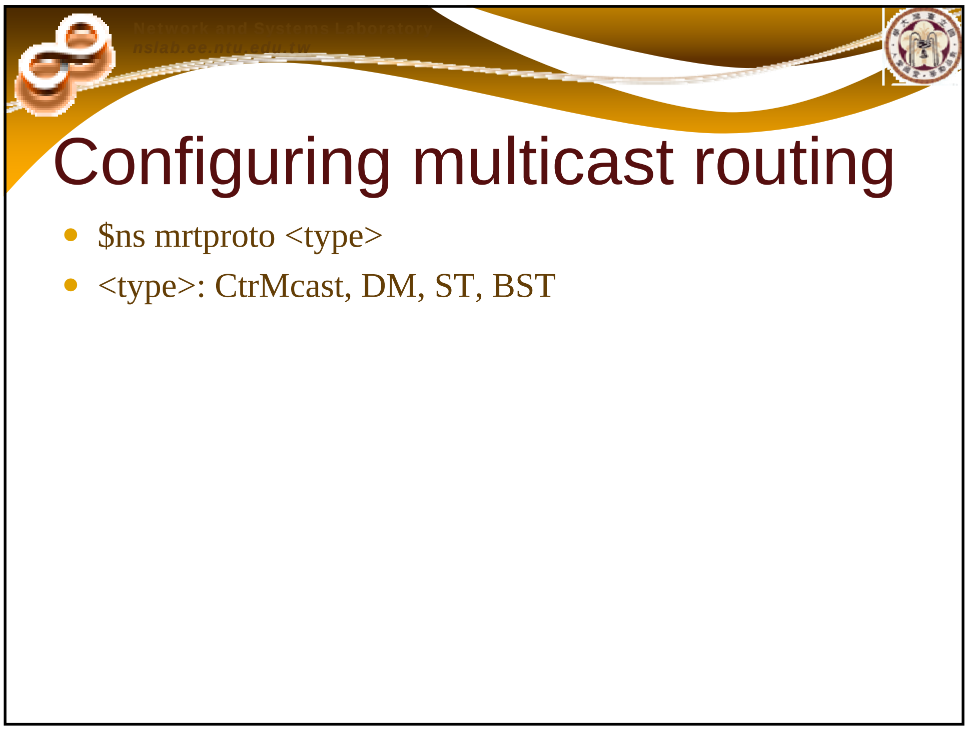
Multicast - 5 components

- enable multicast capability
- configure multicast routing
- create a multicast group/sender
- create a multicast receiver
- attach traffic source



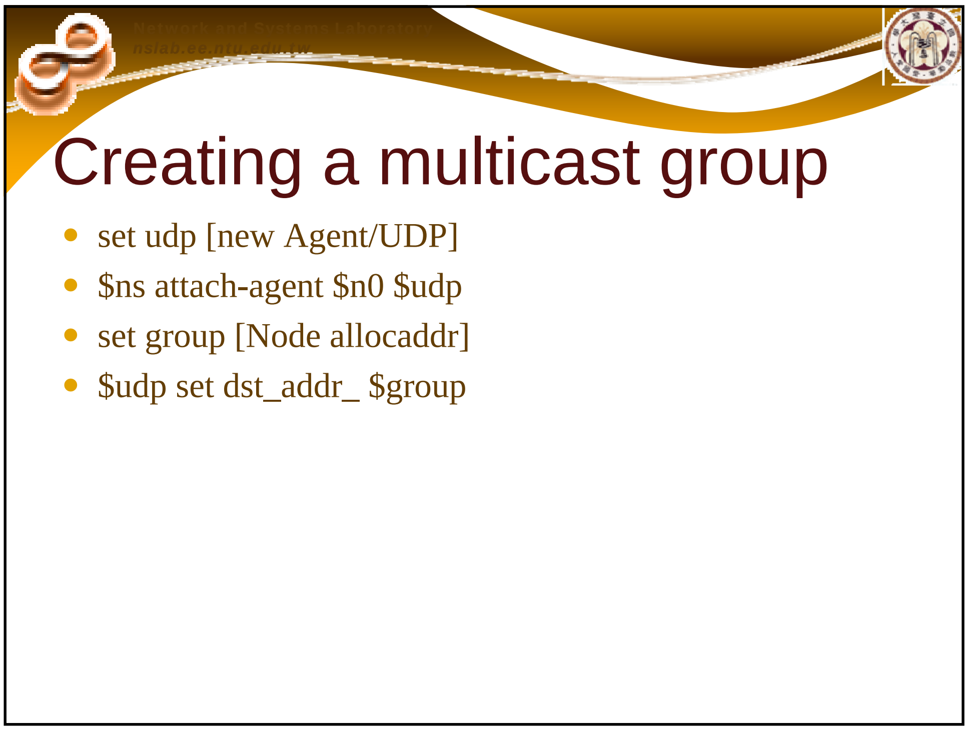
Enabling multicast capability

- set ns [new Simulator -multicast on]
- or $ns multicast (right after [new Simulator])



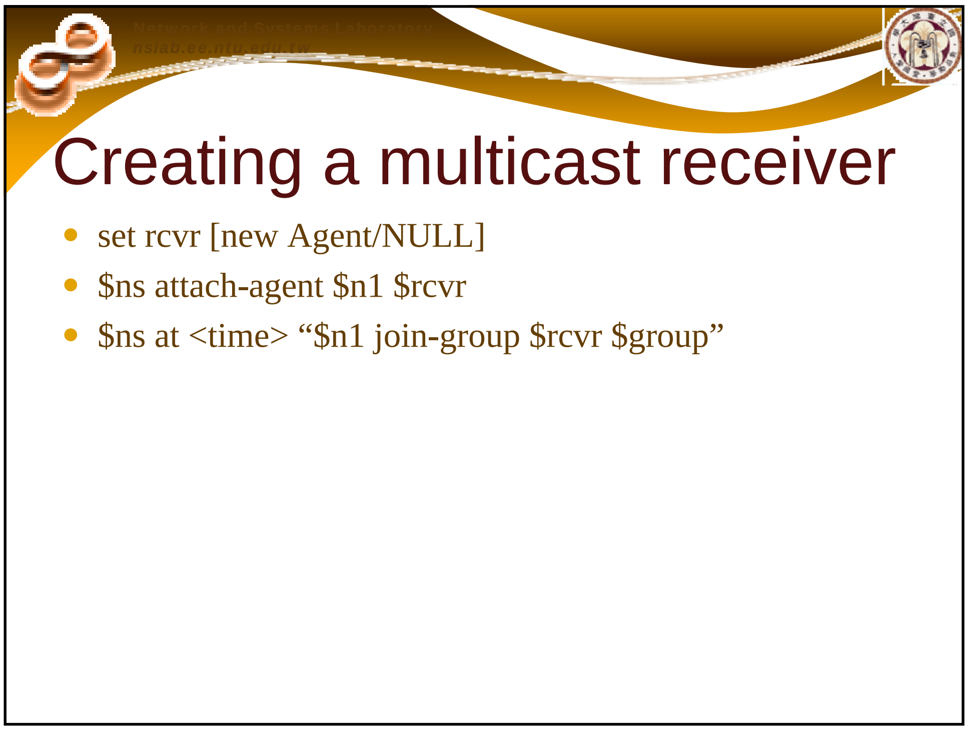
Configuring multicast routing

- $ns mrtproto <type>
- <type>: CtrMcast, DM, ST, BST



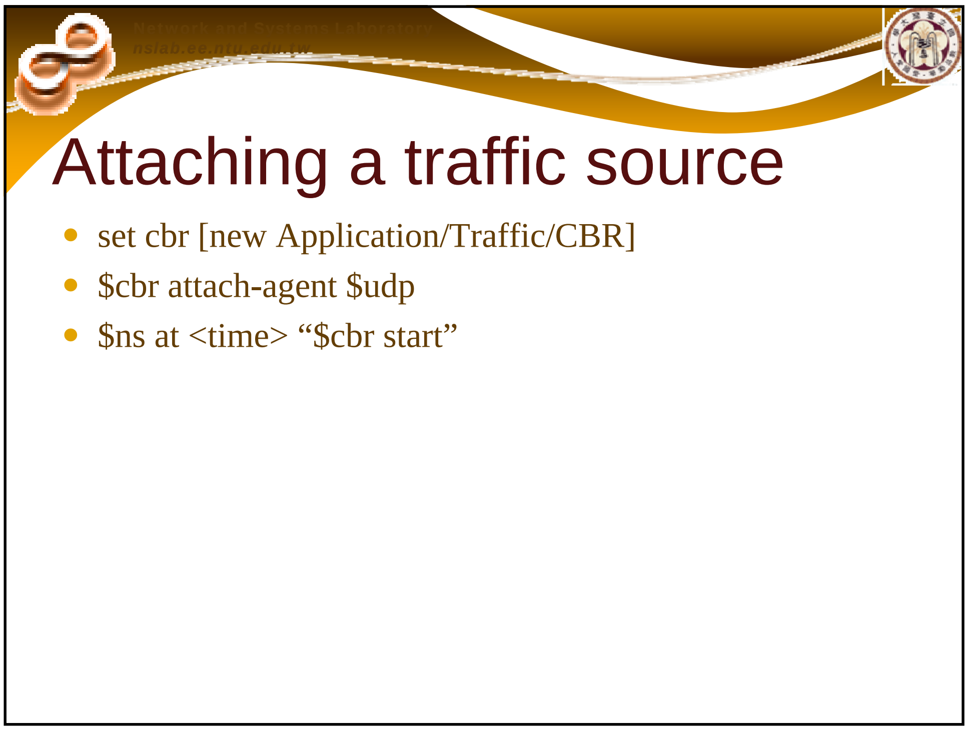
Creating a multicast group

- set udp [new Agent/UDP]
- $ns attach-agent $n0 $udp
- set group [Node allocaddr]
- $udp set dst_addr_ $group



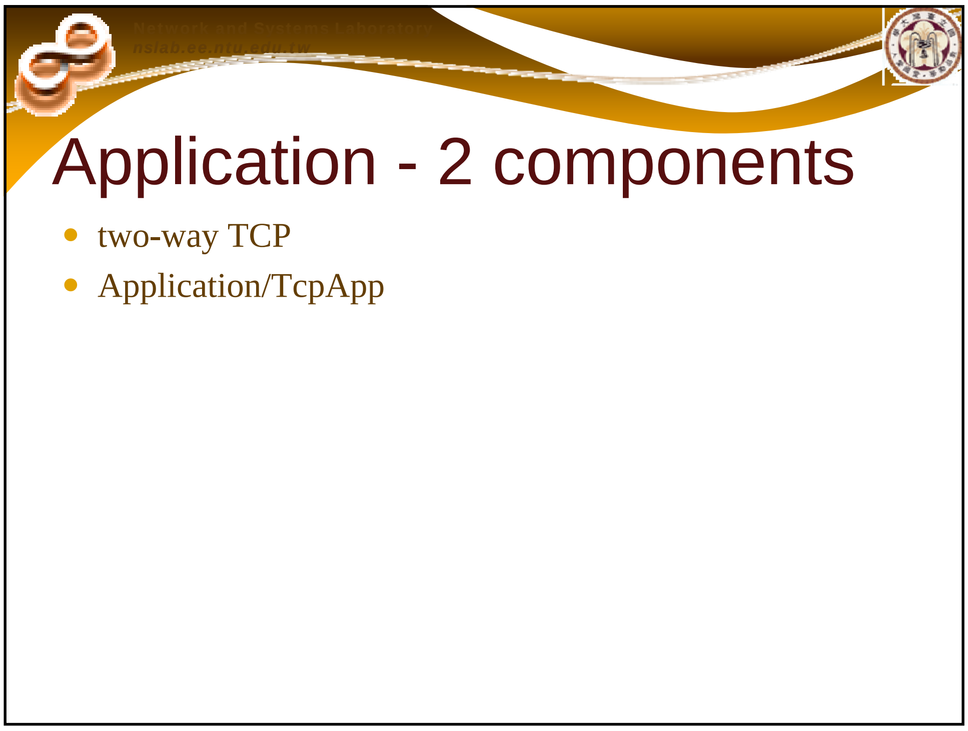
Creating a multicast receiver

- set rcvr [new Agent/NULL]
- $ns attach-agent $n1 $rcvr
- $ns at <time> “$n1 join-group $rcvr $group”



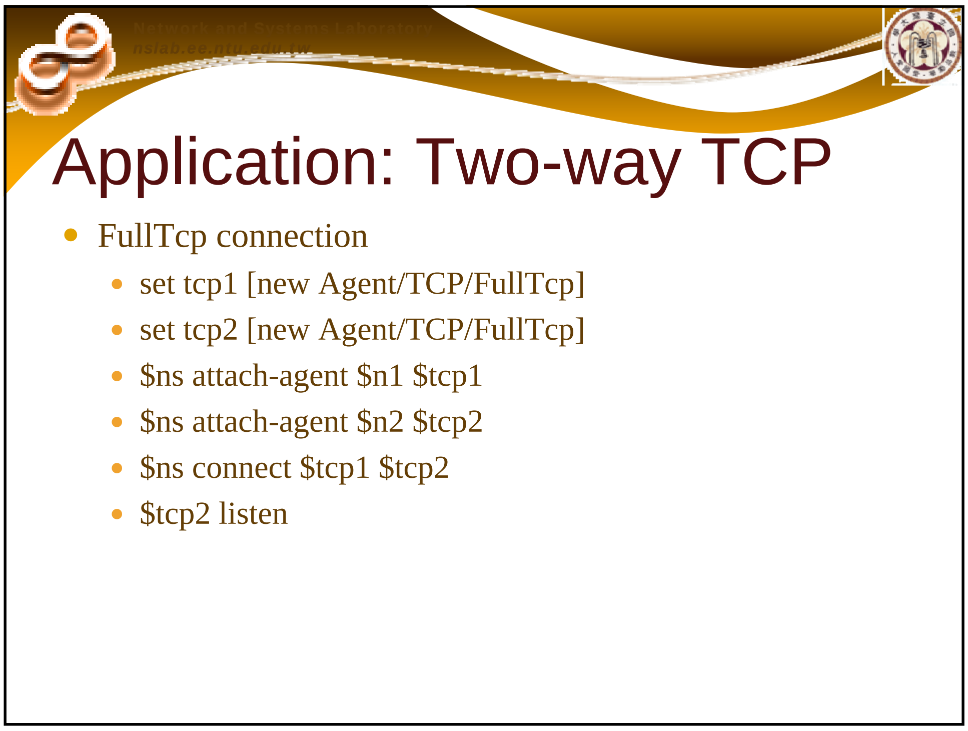
Attaching a traffic source

- set cbr [new Application/Traffic/CBR]
- $cbr attach-agent $udp
- $ns at <time> “$cbr start”



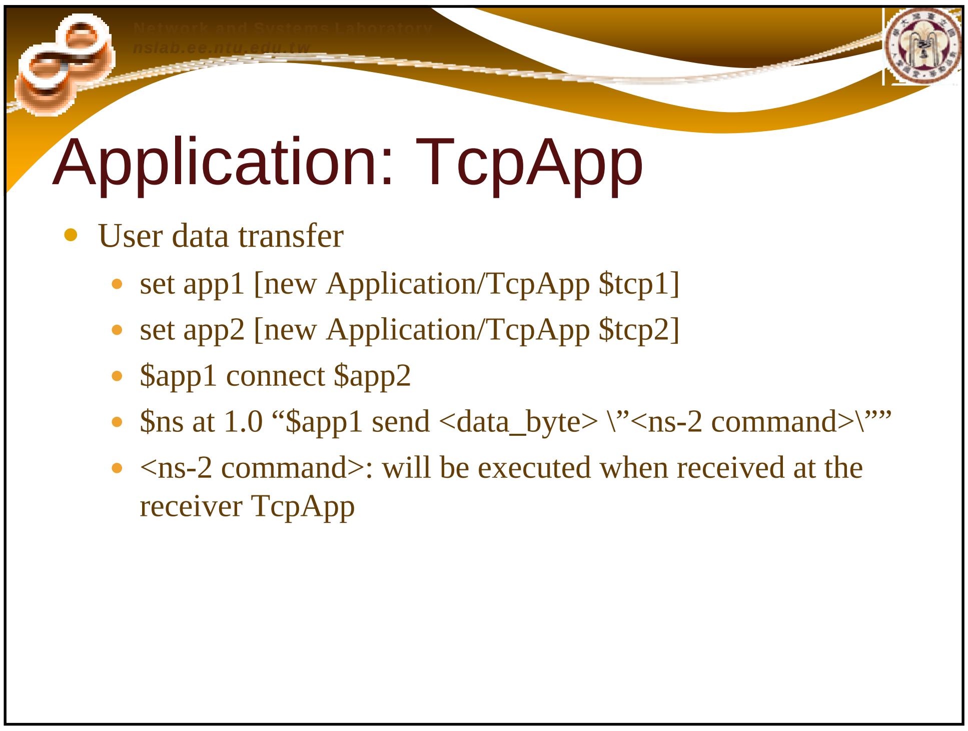
Application - 2 components

- two-way TCP
- Application/TcpApp



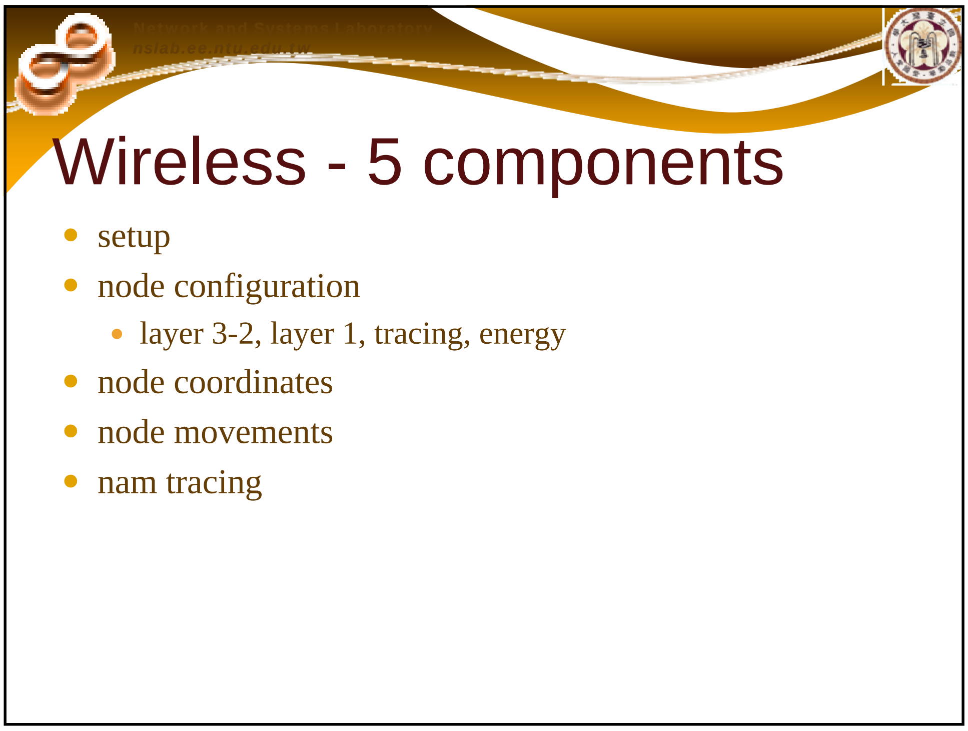
Application: Two-way TCP

- FullTcp connection
  - set tcp1 [new Agent/TCP/FullTcp]
  - set tcp2 [new Agent/TCP/FullTcp]
  - $ns attach-agent $n1 $tcp1
  - $ns attach-agent $n2 $tcp2
  - $ns connect $tcp1 $tcp2
  - $tcp2 listen



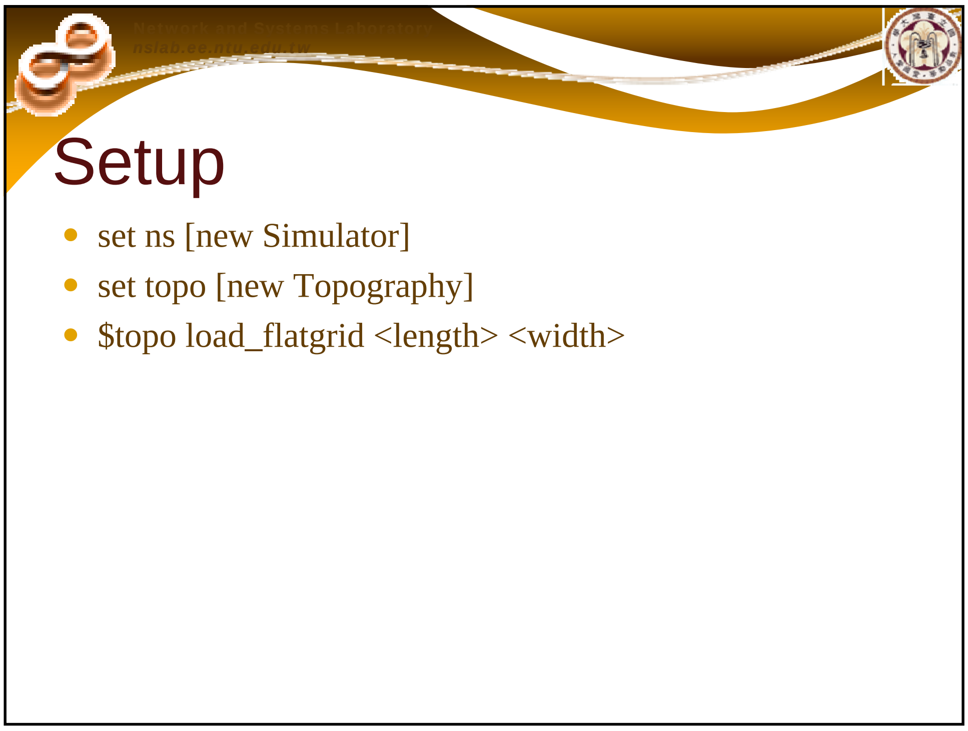
Application: TcpApp

- User data transfer
  - set app1 [new Application/TcpApp $tcp1]
  - set app2 [new Application/TcpApp $tcp2]
  - $app1 connect $app2
  - $ns at 1.0 “$app1 send <data_byte> \”<ns-2 command>\””
  - <ns-2 command>: will be executed when received at the receiver TcpApp



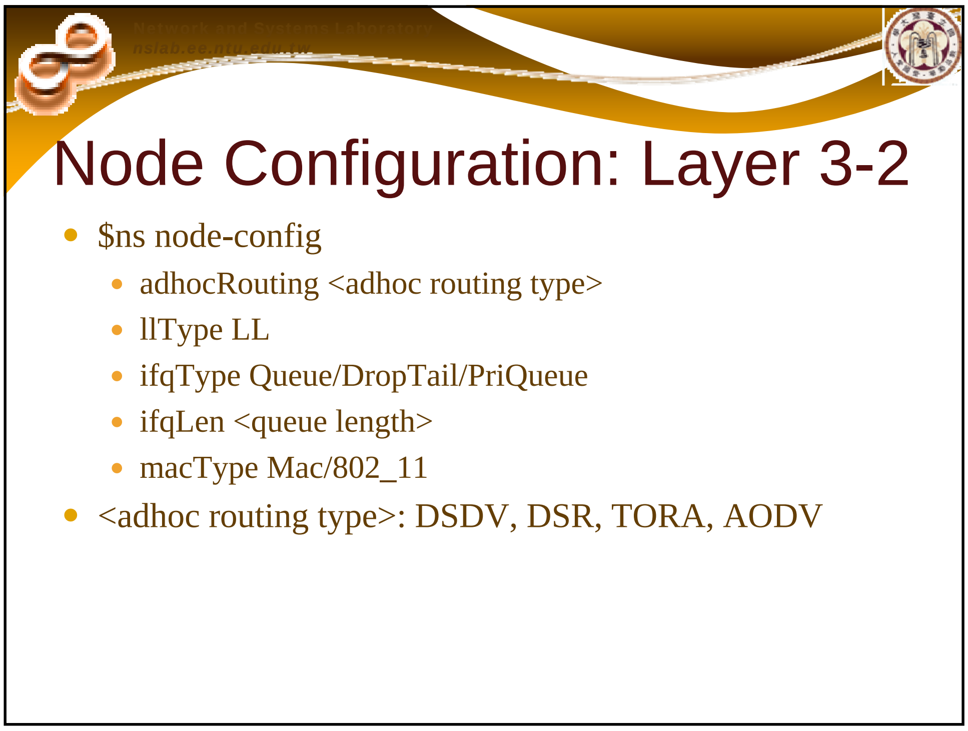
Wireless - 5 components

- setup
- node configuration
  - layer 3-2, layer 1, tracing, energy
- node coordinates
- node movements
- nam tracing



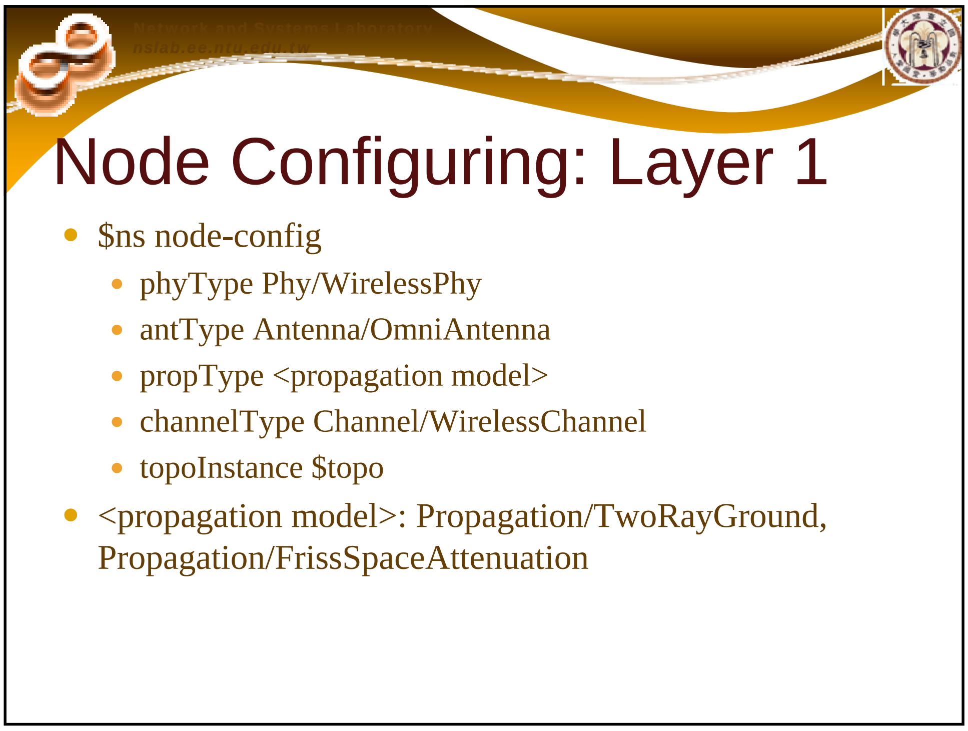
Setup

- set ns [new Simulator]
- set topo [new Topography]
- $topo load_flatgrid <length> <width>



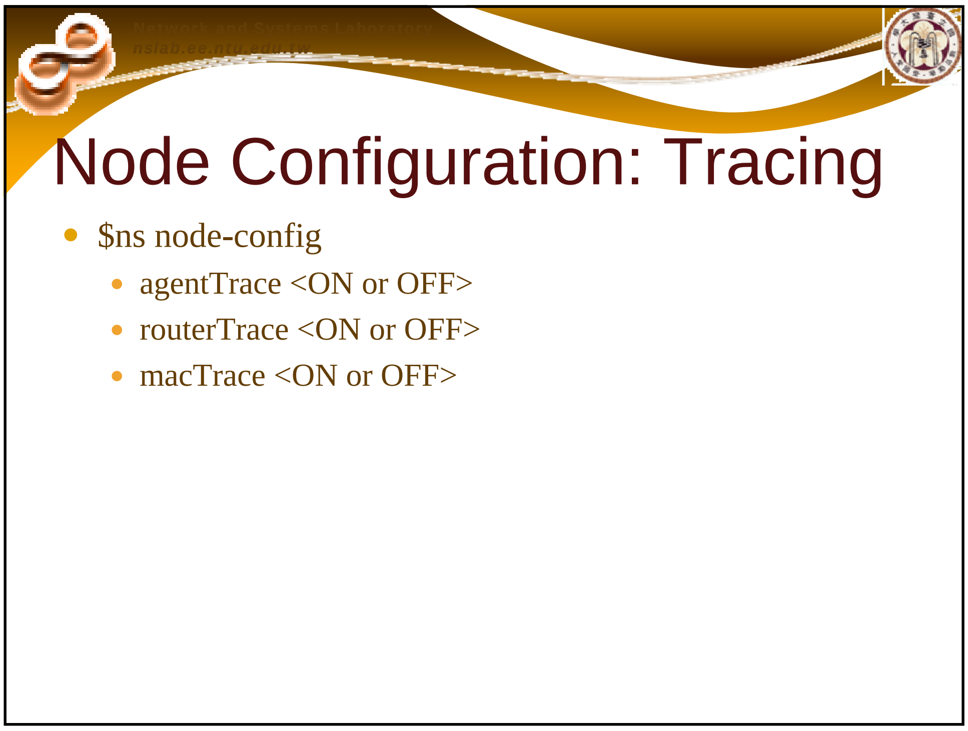
Node Configuration: Layer 3-2

- $ns node-config
  - adhocRouting <adhoc routing type>
  - llType LL
  - ifqType Queue/DropTail/PriQueue
  - ifqLen <queue length>
  - macType Mac/802_11
- <adhoc routing type>: DSDV, DSR, TORA, AODV



Node Configuring: Layer 1

- $ns node-config
  - phyType Phy/WirelessPhy
  - antType Antenna/OmniAntenna
  - propType <propagation model>
  - channelType Channel/WirelessChannel
  - topoInstance $topo
- <propagation model>: Propagation/TwoRayGround, Propagation/FrissSpaceAttenuation



Node Configuration: Tracing

- $ns node-config
  - agentTrace <ON or OFF>
  - routerTrace <ON or OFF>
  - macTrace <ON or OFF>

Node Configuration: Energy

- $ns node-config
  - energyModel EnergyModel
  - initialEnergy <total energy>
  - txPower <energy to transmit>
  - rxPower <energy to receive>

Creating Nodes

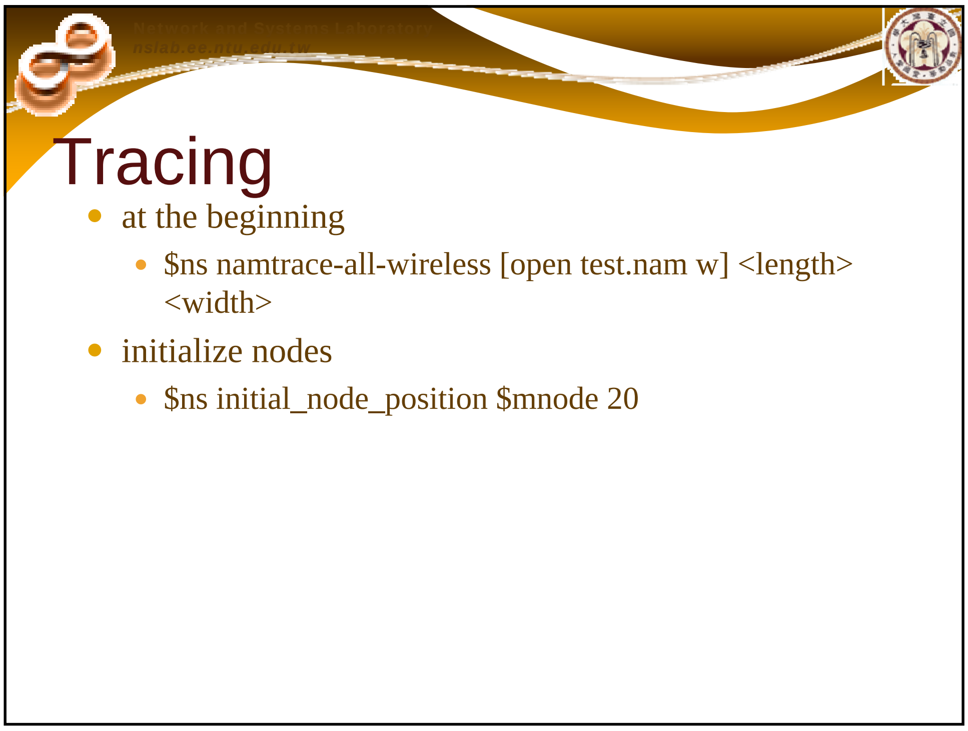
- set mnode [$ns node]

Node Coordinates

- $mnode set X_ <x>
- $mnode set Y_ <y>
- $mnode set Z_ 0

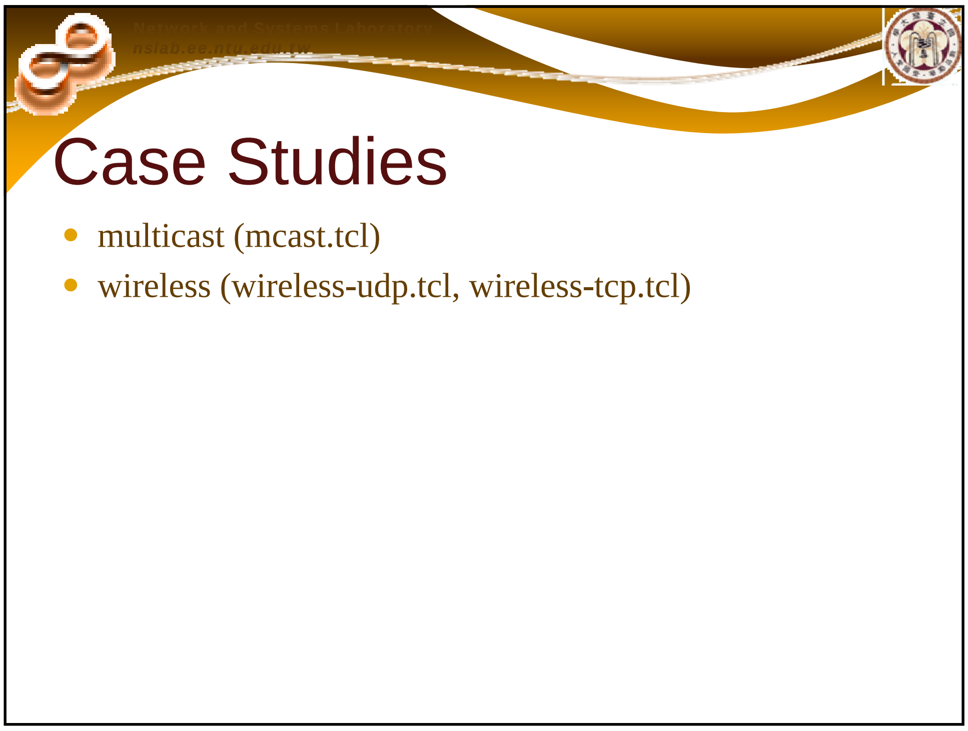
Node Movement

- Disable random motion
  - $mnode random-motion 0
- Specified
  - $ns at 1.0 “$mnode setdest <x> <y> <speed>”
- Random
  - $ns at 1.0 “$mnode start”



Tracing

- at the beginning
  - $ns namtrace-all-wireless [open test.nam w] <length> <width>
- initialize nodes
  - $ns initial_node_position $mnode 20



Case Studies

- multicast (mcast.tcl)
- wireless (wireless-udp.tcl, wireless-tcp.tcl)

Basic ns-2: Not Covered

- mobile IP
- satellite
- DiffServ
- emulation

Lab 1

- We will do it in the EE Lab
  - EEII R132
  - 電腦訓練班教室

Questions?