2nd ISSNSM’s Tutorial on

Simulating Networks with Network Simulator 2 (ns-2)

(Tutorial T2)

Speaker:
Frank Eyermann

June 3, 2008
ISSNM program chaired by Burkhard Stiller, David Hausheer, University of Zürich

ISSNM laboratory organization chaired by Cristian Morariu, Peter Racz, University of Zürich
NS-2 Network Simulator 2

Tutorial – Emanics Summer School, Zurich
3rd June, 2008

This tutorial/training course was supported in part by the EC IST-EMANICS Network of Excellence (#26854).

Introduction

Frank Eyermann
– Frank.Eyermann@unibw.de

– Information Systems Laboratory
– Faculty for Computer Science
– Universität der Bundeswehr, Munich
Exercise 1.1: ns-2’s Hello World

Create the following scenario
- Two nodes, link in between
- Sender agent: Agent/UDP
- Receiver agent: Agent/Null
- Connect agents
- Data source: Application/Traffic/CBR
- Run from 0.5 to 4.5 sec, finish at 5.0 sec

Exercise 1.2

Extend to the following scenario
Exercise 1.2

- Connect the appropriate agents
- Start the FTP application at t = 0.5s
- Start the CBR data source at t = 1s
- Terminate both at t = 4.5s
- Visualize the bottleneck queue:
  $ns duplex-link-op $n2 $n3 queuePos 0.5

☐ Run the simulation
☐ Replay the simulation with nam.
  - Observe the queue and the load on the bottleneck link

Exercise 1.3

☐ Change the bottleneck drop policy
  - from DropTail to RED (Random Early Detect)
  - What changes can you observe?

☐ The FTP traffic is now very bursty.
  - Change the RED queue parameters trying to make it smoother
    - set l [$ns link $n2 $n3]
    - set q [$l queue]
    - #$q set thresh_ ?? # default 0
    - #$q set maxthresh_ ?? # default 50
    - #$q set linterm_ ?? # default 0.1
Exercise 1.4

- Based on 1.2
- Name each flow
  - \$agent set fid\_number
- Color flow 1 red and flow 2 blue
  - \$ns color number Color
- Predefine the links’ orientation for better layout
  - \$ns duplex-link-op \$n0 \$n1 orient right-down
- Restart simulation

Exercise 1.5

- Based on exercise 1.3
  - Despite tuning the queue’s parameter the queue length changes in drastically
  - Color the packets
  - Do you see why the RED queue does not perform as expected?
  - What function needs to be implemented?
Exercise 2.1: Tracing

- Based on exercise 1.4
- Trace all events on the bottleneck link
- Create a second trace file for only the drops
- Open the document „NS-2 Trace Formats“
  - What is the normal trace format?

Exercise 2.2: Monitoring

- Based on Exercise 1.4
- Create a monitor for the central node
- Use sampleInterval=0.1 to periodically write information to file
- Do not forget to trigger the monitoring
Exercise 2.3: Monitoring

- Based on Exercise 1.4
- Create a monitor for the central node
- Read all 0.05 sec the following values for each time period:
  - Throughput
  - Packet drops
  - Average packets in queue
  - Average time of a packet in queue
- Write them to a file

Hints:
- You need to create a Samples object for the average delay
- Create a procedure which retrieves the information from the Monitor and writes them to file
- The average packets in queue you get from the respective built-in integrator
- Care for the case when no packets have been received in one period
- Have the procedure schedule itself again at $[expr \text{[ns now]} + 0.05]$
Exercise 2.4: Monitoring

- Based on exercise 1.2 and 1.3
- Compare utilization of the bottleneck link and average time of a packet in the queue for DropTail and RED queuing discipline
  - Use QueueMonitors and Samples for the respective data

Exercise 2.5: Monitoring Flows

- Based on exercise 2.2
- Monitor the both flows separately
  - Use the standard output format
  - Look for the format at page 243 of ns_doc.pdf
Exercise 3.1: Trace Post-Processing

- Based on exercise 2.3
- Visualize the drops and the average packets in the queue with xgraph and gnuplot

---

Exercise 3.2

- Create the following scenario
- All clients pull FTP data from 0.5 to 4.5 sec
Exercise 3.2

- Should Client 1 invest in upgrading to a 25 MBit/s or 50 MBit/s link?
- Or better invest in a Fast-Path option reducing the delay to 15ms?
- Could the provider’s queuing discipline influence the user’s decision?

Exercise 3.3

- Based on exercise 1.4
- Visualize the packet jitter of the CBR data
  - Remember to skip dropped packets
Exercise 4.1: LANs

- Create a typical 10 MBit/s Ethernet
  - FTP-Server ➔ Router ➔ 4 FTP-Clients

- The LAN delay should be 1 ms
- The connection from the server to the router is a WAN (2mbit/s, 10 ms)
- Simulate 5 sec. FTP-Transfer

Exercise 4.2

- nam shows a quite long queue
- What is the average queuing delay of a packet?
- How could the delay be reduced? Think of exercise 1.3
  - Repeat the simulation
Exercise 5.1: Routing

- Create the following scenario, all links 10MBit, 8 ms, DropTail
- Send CBR data
  - 1 -> 6, 3 -> 8,
  - 5 -> 2, 7 -> 4
- Rate = 1.5 MBit, Size = 500 Byte
- Activate session-based routing

Exercise 5.2

- Link node 1 to node 2
  - Fails each 1.5 sec for 0.5 sec
- Node 5
  - Is down between 1.6 and 2.5 sec
Exercise 5.3: Routing

- Based on exercise 4.1
- Dialup backup line, if WAN connection fails
  - Volume based, quite expensive
  - 1mb, 15ms
  - Simulate deterministic failure 2sec up, 0,5sec down

```
foreach node [[Simulator instance] all-nodes-list]
{
    # XXX using dummy 0 for 'changes'
    $node notify-mcast 0
}
```

- "LAN" is a node (LanNode) but misses notify-mcast
- Hack: find method notify-mcast of node
- Create an (empty) method stub for LanNode
- Run make in folder ns-2.33