S-38.180 – Quality of Service in Internet

Introduction to the Exercises

Timo Viipuri

8.10.2003

Exercise Subjects

1) General matters in doing the exercises
   - Work environment
   - Making the exercises and returning the reports
2) Introduction to NS-2 Network Simulator
   - Basic understanding on how to work with it

Work Environment

- Class rooms: Maari-c and Maari-d
  - http://www.hut.fi/atk/luokat/maari-c.html (Linux)
  - http://www.hut.fi/atk/luokat/maari-d.html (Windows)
- Linux OS
  - Beginners Guide:
    - http://www.redhat.com/docs/manuals/linux/RHL-9-Manual/getting-started-guide
    - http://linux.org.mt/article/terminal
  - Command Reference:
    - http://linux.nixcraft.com/linux_commands

Exercises

- Exercise schedule and material:
  - http://www.netlab.hut.fi/opetus/s38180/s03/schedule.shtml
- Each exercise session (2 hrs) consists of:
  - (Review of the previous exercise)
  - Introduction to the new exercise
  - Begin work on the simulations with course staff present
- Do all the exercises in the Computing Centre's computers
  - The NS-2 software found there is not the standard distribution -> some exercises won't work elsewhere

Exercise reports

- Hard deadline for all reports is **October 29th, 4 pm**
  - It is advised to return reports before the next exercise
  - Return format is either **PDF** or **paper**
- Two types of grading depending on the exercise:
  1. Fail / Pass or
  2. Fail / Satisfactory / Good / Excellent
- All exercises must be passed to complete the course
- Exercise points are summed up and scaled to 1-6
  - Used in the exam grading to replace the points from the lowest scoring answer

S-38.180 – Quality of Service in Internet

Exercise 1: NS-2 Network Simulator

Timo Viipuri

8.10.2003

Exercise Objectives

- To familiarize yourself with the work environment
- To learn to work with NS-2 at the level that you can:
  1. Write simple simulation scripts
  2. Read and understand more complex simulation scripts

Tasks of the Day

1. A few words about the background and structure of NS-2
   - to give you some idea of what you are working with
2. Line-by-line study of a simple simulation scenario
   - to explain the minimum requirements needed to create a simulation
3. Begin making your own simulation
   - to get a hands-on feeling on the simulator and prepare you for the later exercises

NS-2 Forewords

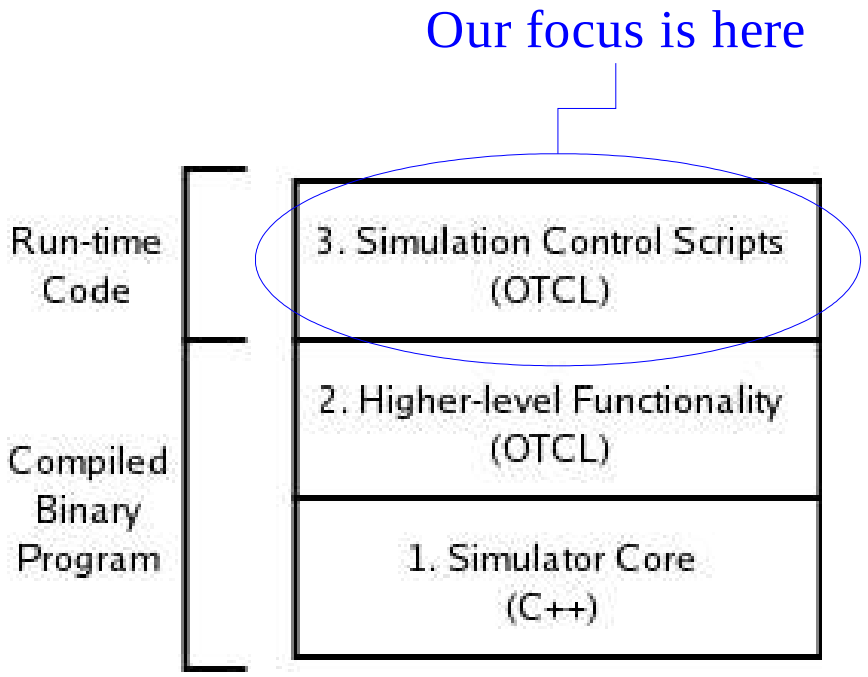
- Began as a variant of the REAL network simulator in 1989
- Open source software
  - Possible to tailor the code to exactly fit the needs
  - Thousands of developers => rapid increase in functionality
- Nowadays it is argueably the most popular network simulation tool in the world
  - Used extensively by both businesses and universities

NS-2 Software Structure

- NS-2 uses two programming languages to combine efficiency and ease of extentability
  - C++
  - OTCL (Object Tool Command Language)
- NS-2 software is written in both C++ and OTCL
  - Generally doesn't need to be modified
- Simulation scripts are written in OTCL
  - Used to setup and control the simulation

NS-2 Software Structure 2

- Simulator software is separated to 3 layers:
  1. Basic functionality: C++
  2. Experimental protocols and complex applications: OTCL
  3. Simulation control scripts: OTCL

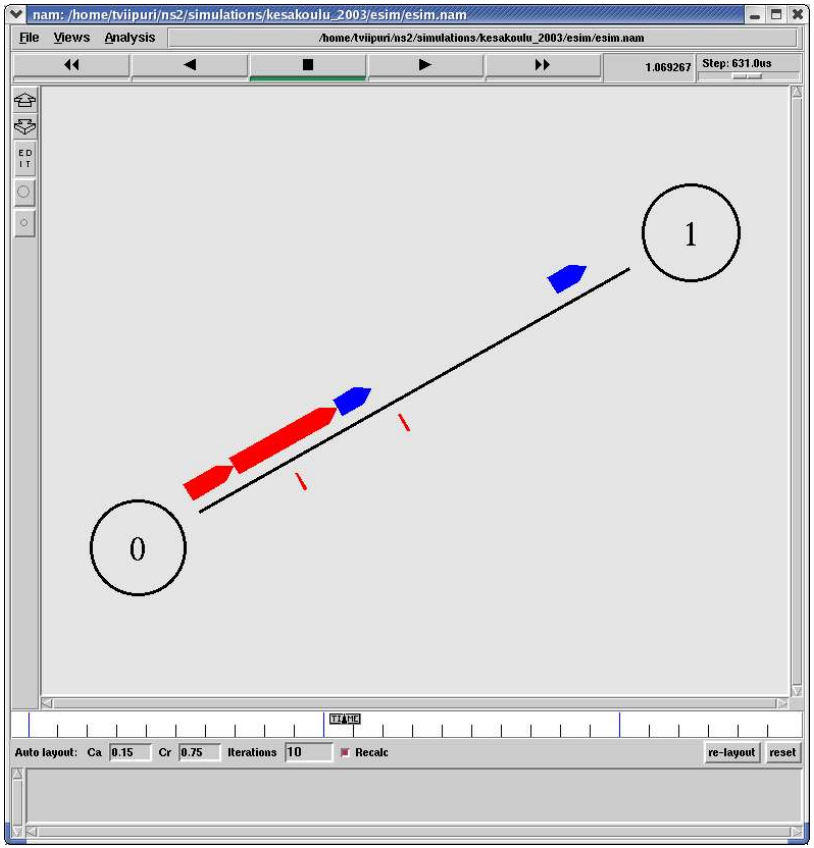


Simulation Scripts

- Used to set-up a simulation scenario
  - Network topology
  - Traffic agents
  - Simulation events, e.g. when to start sending data
  - Gathering results: monitoring and tracing
- Written in OTCL
  - No need to compile; scripts are interpreted at run-time

NAM – Network Animator

- Animation tool for graphically viewing simulation results
- Useful for examining simple simulations



Simulation Example

- Topology
  - A network of two nodes connected with a duplex link
    - Bandwidth: 5 Mbps
    - Packet delay: 10 ms
- Traffic agents
  - 1 TCP-connection
  - 1 UDP-connection with a CBR-traffic generator
- Simulation events
  - TCP starts sending 15 kB of data at 0.5 s
  - UDP starts sending at a rate of 800 kbps at 0.2 s and stops at 0.8 s
- Gathering data
  - Trace all packet events

Example 2: Topology

- Create nodes n0 and n1

set n0 [$ns node]

Create a node and assign it to variable n0

Assign a variable n0

set n1 [$ns node]

- Create a duplex-link between the nodes

$ns duplex-link $n0 $n1 5Mb 10ms DropTail

Call procedure 'duplex-link' of object $ns

Set link between nodes n1 and n2

Bandwidth 5Mbps, delay 10ms

Buffer management method: DropTail

Example 3: UDP-agents

- Create UDP- and null-agents

  set udp0 [new Agent/UDP]

  set null0 [new Agent/Null] — A null-agent acts as an UDP-sink

- Attach them to nodes n0 and n1

  $ns attach-agent $n0 $udp0 — Parameters: $node $agent

  $ns attach-agent $n1 $null0

- Connect the agents

  $ns connect $udp0 $null0 — Parameters: $agent $agent

Example 4: CBR-traffic

- Create a CBR traffic source

  set cbr0 [new Application/Traffic/CBR] — Application type

- Set traffic parameters

  $cbr0 set packetSize_ 500

  $cbr0 set interval_ 0.005 — Time interval between packets

$$\Rightarrow \text{SendRate} = \frac{8 * 500\ \text{b}}{0.005\ \text{s}} = 800\,\text{kbps}$$

- Attach the traffic generator to an agent

  $cbr0 attach-agent $udp0

Example 5: TCP-agents

- Create a TCP-connection pair

  set clnt0 [new Agent/TCP/FullTcp]

  set srvr0 [new Agent/TCP/FullTcp]

  FullTcp includes a three-way handshake and tear-down

- Attach agents to nodes

  $ns attach-agent $n0 $srvr0

  $ns attach-agent $n1 $clnt0

- Connect the agents

  $ns connect $srvr0 $clnt0

- Assign the client-agent to listening mode

  $clnt0 listen

Example 6: Events

- Schedule events

$ns at 0.2 "$cbr0 start"

Start sending CBR-data

Launch the quoted command at 0.2 s

$ns at 0.5 "$srvr0 sendmsg 15000 \"MSG_EOF\""

Send 15 kB of TCP-data

$ns at 0.8 "$cbr0 stop"

Stop sending CBR-data at 0.8 s

- Call the finish procedure after 1.0 s of simulation time

$ns at 1.0 "finish"

- Start the simulation in the end of the script

$ns run

Example 7: Tracing

- Open files for writing

set nsf [open example.ns w]

Open the file for writing

File handle in the simulation

Name of the file

set namf [open example.nam w]

- Set trace types

Output file handle

$ns trace-all $nsf

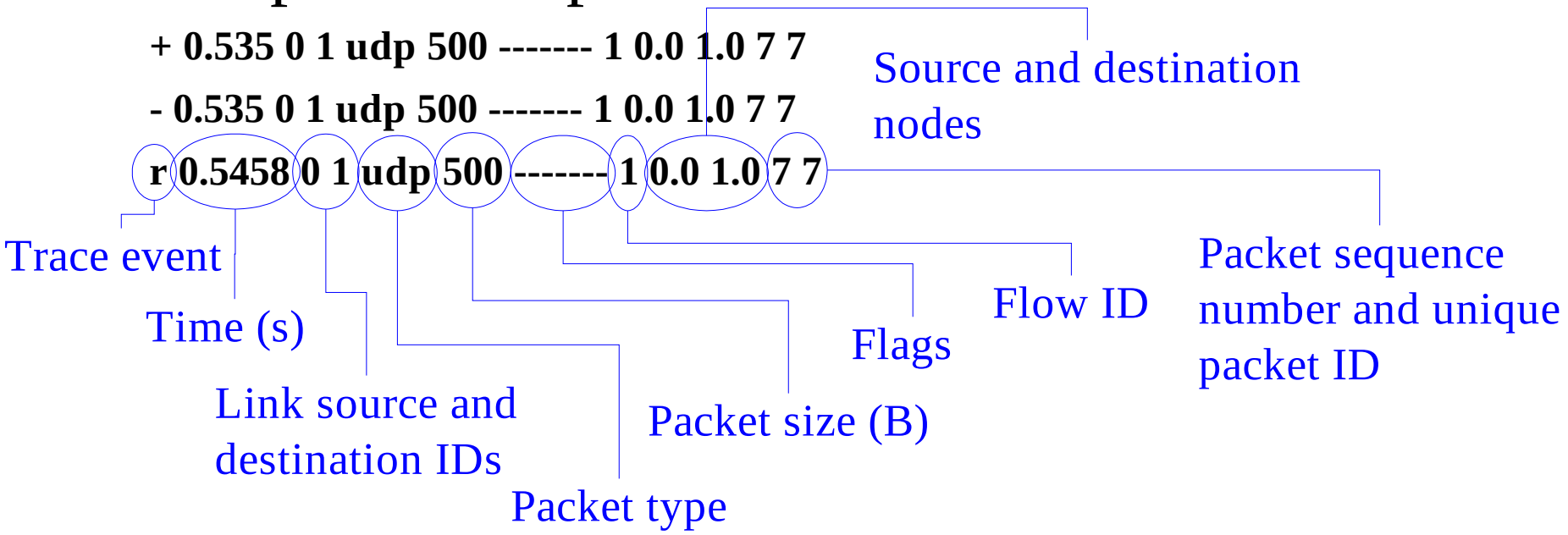
Trace all links

$ns namtrace-all $namf

Trace all links for NAM (Network Animator)

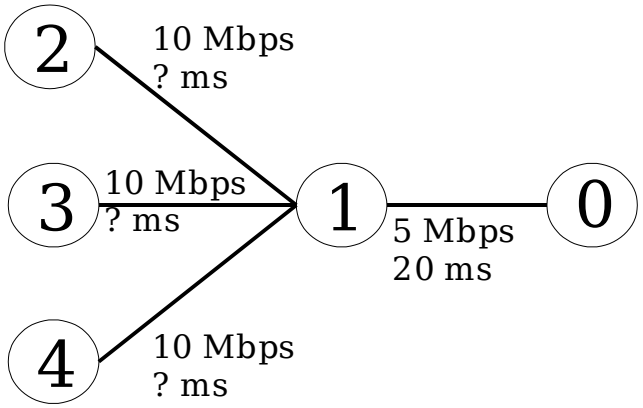
Example 8: Results

- Sample of the packet trace file:



Simulation: Link Delay

- Topology
  - 1 FTP client
    - Node 0
  - 3 FTP servers
    - Nodes 2-4



- Study the effect of link delay to the throughput of a TCP-connection

Random Numbers

- NS-2 produces only pseudo-random numbers
  - they aren't random but only appear to be
- A seed value is needed for the generation of pseudo-random numbers
  - If the seed value is the same the number sequence will be the same
- In NS-2 the seed value is modified with:

  "$defaultRNG seed 1",
  - using seed 0 will cause a random seed to be generated on each new simulation
- e.g. RED uses random numbers to calculate the drop probability

NS-2 Material

- Development pages:
  - http://www.isi.edu/nsnam/ns
  - Especially useful topics:
    - "Ns manual"
    - "Mark Greis's tutorial"
  - Visit them!
- TCL tutorials
  - http://users.belgacom.net/bruno.champagne/tcl.html
  - http://hegel.ittc.ukans.edu/topics/tcltk/tutorial-noplugin
- OTCL tutorial
  - http://nestroy.wi-inf.uni-essen.de/Lv/gui/otcl