Network Simulation with ns-3

Outline

L3.1  Motivation and overview
L3.2  ns-3 installation and use
L3.3  Network simulation example
Network Simulation with ns-3

Motivation and Overview

L3.1 Motivation and overview
L3.2 ns-3 installation and use
L3.3 Network simulation example

Motivation and Overview

Network Analysis Techniques

• Analytical analysis
  – mathematical analysis/modeling of systems (e.g. RTT calc.)
• Simulation
  – model the system at abstract level via software
  – various network simulators exist (e.g. ns-3, OPNET, OMNet)
• Emulation
  – HW component that behave like real system (e.g. ONL)
• Measurement
  – active (e.g. ping, traceroute) or passive (e.g. Wireshark)
• Experimentation
  – experiment on a testbed (e.g. GENI)
Motivation and Overview

Network Simulation

- **Goal**
  - build software *simulation model* of system
  - to analyze/study/improve/develop network protocols
- **Reason**
  - real systems are expensive, complex, unavailable
- **Advantages**
  - relatively easy and less time consuming
- **Disadvantages**
  - simplified view of complex interactions
  - could be immensely misleading
    - dependence on assumptions and model

Motivation and Overview

Simplified Network Simulation Flowchart

- Define topology, protocols, models
- Specify initial values
- Process output/trace files
- Plot throughput, goodput, delay etc.
- Are results expected?
  - Confidence level attained?
Motivation and Overview

ns-3 Highlights and History

• ns-3 is a discrete-event network simulator for:
  - Internet systems
    • emphasis on layer 2-4
  - targeted for research and education
  - aims to replace venerable ns-2 simulator
• Community-oriented open source development
  - integration of open source tools/software
    • e.g. Wireshark, tcpdump
• First release in June 2008
  - latest release (ns-3.7.1) on January 2011
  - planned for new releases every 3-4 months

Motivation and Overview

Architecture, Protocols, Models of ns-3

• ns-3 software architecture is built on C++
  - Python for front-end (e.g. scripting, visualization)
• Some protocols/models include:
  - socket like API, on/off application
  - TCP, TCP stack emulation (Linux, BSD), UDP
  - IPv4, IPv6 support, Static routing (Dijkstra)
  - MANET (OLSR, AODV, DSDV*)
    • * by Hemanth Narra & Yufei Cheng supervised by Dr. Sterbenz
  - IEEE 802.11 and variants, PPP
  - IEEE 802 physical layers, Rayleigh fading channel
  - mobility models: 3D Gauss-Markov (merged to ns-3.8)
    • developed by Dan Broyles under supervision of Dr. Sterbenz
Network Simulation with ns-3

ns-3 Installation and Use

L3.1 Motivation and overview
L3.2 ns-3 installation and use
L3.3 Network simulation example

ns-3 Installation and Use

Installation Overview

- ns-3 can be installed on common platforms
  - desktop & servers - 32 bit & 64 bit architectures
  - any major OS: Linux, Mac OS, Windows
    - Windows requires Cygwin
    - we do not support ns-3 in Cygwin, use virtual machine instead
  - detailed instructions:
    - http://www.nsnam.org/getting_started.html

- Installation includes following steps:
  - download
  - build
  - validation
ns-3 Installation and Use

Installation Steps Using waf

- Download ns-3 package, unzip and untar it
  - http://www.nsnam.org/download.html
  - tar -jxf ns-3.x.tar.bz2
- Check the system for prerequisites and build
  - change directory to ns-3.x
  - ./waf -d debug configure
  - ./waf
- Validate build by running unit and regression tests
  - ./waf check
  - ./waf --regression

ns-3 Installation and Use

Installation Notes

- At the end of installation run following and verify:
  - ./waf --run hello-simulator
  - Hello Simulator
- Source code can be downloaded from a repository
  - Mercurial
- There are other methods of building the ns-3
  - using Python script to install downloaded source code
- If you need to uninstall ns-3 package
  - rm -rf ns-3.x
  - where ns-3.x is the directory
ns-3 Installation and Use

Usage Overview

- Decide what you want to simulate
  - define the topology
  - create nodes, channel, network interfaces
  - configure Internet stack and applications
  - set attributes
- Build the simulation script using a text editor
  - e.g. emacs, vi, textpad
- Execute the .cc program via waf
- Analyze output
- Good documentation always helps!

Abstractions

- Simulations performed on an abstract model
- Abstracts represented in C++ by classes
- Classes provide methods to manage representations
- Key objects in ns-3
  - node
  - application
  - channel
  - net device
  - topology helpers
ns-3 Installation and Use

Script Structure

- C++ scripts include the following structure
  - boilerplate: important for documentation
  - module includes: include header files
  - ns-3 namespace: global declaration
  - logging: optional
  - main function: declare main function
  - topology helpers: objects to combine distinct operations
  - applications: on/off application, UdpEchoClient/Server
  - tracing: .tr and/or .pcap files
  - simulator: start/end simulator, cleanup

ns-3 Installation and Use

waf

- waf is a general purpose build system to:
  - configure
  - compile
  - install
- Instead of ./configure;make type ./waf
- waf is Python based
- More information can be found on
  - http://code.google.com/p/waf/
ns-3 Installation and Use

Post-Processing

• Once the simulations are over process trace files
  • Trace files can be filtered via a script
    - e.g. Python, Perl
  • Filtered results can be processed via a plotting tool
    - gnuplot
    - gpwrapper (developed by Abdul Jabbar at KU)
  • Output files in .pcap format is possible
    - Wireshark or tcpdump can be used to view .pcap files
• Logs can be enabled to analyze output
• ns-3 package built-in tools for post-processing
  - flow monitor

ns-3 Installation and Use

Troubleshooting

• Internal workings can be viewed by log output
  - export 'NS_LOG=*=level_all|prefix_func|prefix_time'
  - ./waf --run scratch/first >& log.out
• For C++ you can use gdb tool for debugging
  - ./waf -shell
  - gdb ./build/debug/scratch/my-app
• Memory debugging via valgrind tool
  - ./waf --valgrind --regression
Network Simulation Example

Network Simulation Setup

- Two nodes, one network interface device per node
- Point-to-point link
  - transmission delay: 2 ms, data rate: 5 Mbps
- Application
  - UdpEchoClient on node 0, UdpEchoServer on node 1
  - payload size of 1024-byte packet
  - time interval between packets is 1 s
Network Simulation Example

First.cc Script

- **Main function**
  - int main (int argc, char *argv[]) {

- **NodeContainer class, create method**
  - NodeContainer nodes;
  - nodes.Create (2);

- **PointToPoint helper, set link attributes**
  - PointToPointHelper pointToPoint;
    pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
  - pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));

First.cc Script

- **NetDevice container**
  - NetDeviceContainer devices;
  - devices = pointToPoint.Install (nodes);

- **InternetStackHelper**
  - InternetStackHelper stack;
  - stack.Install (nodes);

- **Ipv4AddressHelper**
  - Ipv4AddressHelper address;
  - address.SetBase ("10.1.1.0", "255.255.255.0");
  - Ipv4InterfaceContainer interfaces = address.Assign (devices);
Network Simulation Example

**First.cc Script**

- **Application**
  - `UdpEchoServerHelper echoServer (9);`
  - `ApplicationContainer serverApps = echoServer.Install (nodes.Get (1));`
  - `serverApps.Start (Seconds (1.0));`
  - `serverApps.Stop (Seconds (10.0));`
  - `UdpEchoClientHelper echoClient (interfaces.GetAddress (1), 9);`
  - `ApplicationContainer clientApps = echoClient.Install (nodes.Get (0));`
  - `clientApps.Start (Seconds (2.0)); clientApps.Stop (Seconds (10.0));`

**First.cc Script**

- **Simulator run of scheduled events**
  - `Simulator::Run ();`
- **Destruction of objects and cleanup**
  - `Simulator::Destroy ();`
  - `return 0;`
Network Simulation Example

Post-processing with Wireshark

- pcap traces can be viewed by Wireshark
- For flow diagram: statistics → flow graph

Network Simulation with ns-3

Assignment Configuration

- 2 nodes, with 1 interface each
- Point-to-point link
  - data rate: 1 Mbps, transmission delay: 1 ms
- IP address assignment
  - 192.168.10.0/24
- Application
  - UdpEchoServerServer on port 53
  - packet size: 512 byte
- Enable ASCII and pcap tracing
- Rest of the attribute values: use from the example
Assignment Submission Guidelines

- Write 1–2 page summary
- Report should include the following sections:
  - experiment setup and procedure (topology, issues, etc.)
  - results
    - include flow diagram in Wireshark as screenshot in report
  - conclusions (what you learned, etc.)
- You can discuss with other students but ...
- ... everyone must submit individual report
- Attach .cc file along with your submission
- Send report in PDF format to GTA, cc: Dr. Sterbenz

Extra Credit

- Use matrix-topology example in the folder:
  - ../ns-3.10/examples/matrix-topology
- Generate a topology with at least three nodes
  - do not use the default matrix
- Place nodes of your choice
- Generate NetAnim screenshot
- Extra more credits for identifying bugs in that code
Network Simulation with ns-3

Further Reading

- ns-3 main page (for documents, news, announcements)
  http://www.nsnam.org/
- ns-3 wiki (howtos, roadmap)
  http://www.nsnam.org/wiki/index.php/Main_Page
- ns-3 documentation: tutorial (chapter 4, 5, 6), manual, doxygen
  http://www.nsnam.org/docs/release/3.10/tutorial/singlehtml/
- ns-3 users mailing list (usage, implementations, discussions)
  http://groups.google.com/group/ns-3-users
- ns-3 bug list (closed, open bugs)
  http://www.nsnam.org/bugzilla/

Acknowledgements

Some material in these foils comes from the ns-3 tutorial presentations from conferences, workshops:

- Tom Henderson,
  ns-3 tutorial
  SIMUTools 2009
  http://www.nsnam.org/tutorials.html
- Gustavo Carneiro,
  NS-3 Tutorial
  April 2010
  http://www.nsnam.org/tutorials/NS-3-LABMEETING-1.pdf
Network Simulation with ns-3

Other References

• C++ tutorials online
  - and many more links and books on the subject
• GDB
  - http://www.gnu.org/software/gdb/
• valgrind
  - http://valgrind.org/
• gnuplot
  - http://www.gnuplot.info/
• Python
  - http://www.python.org/