Quick NS-3 Tutorial

Serhat Arslan, Jenya Pergament
April 2021

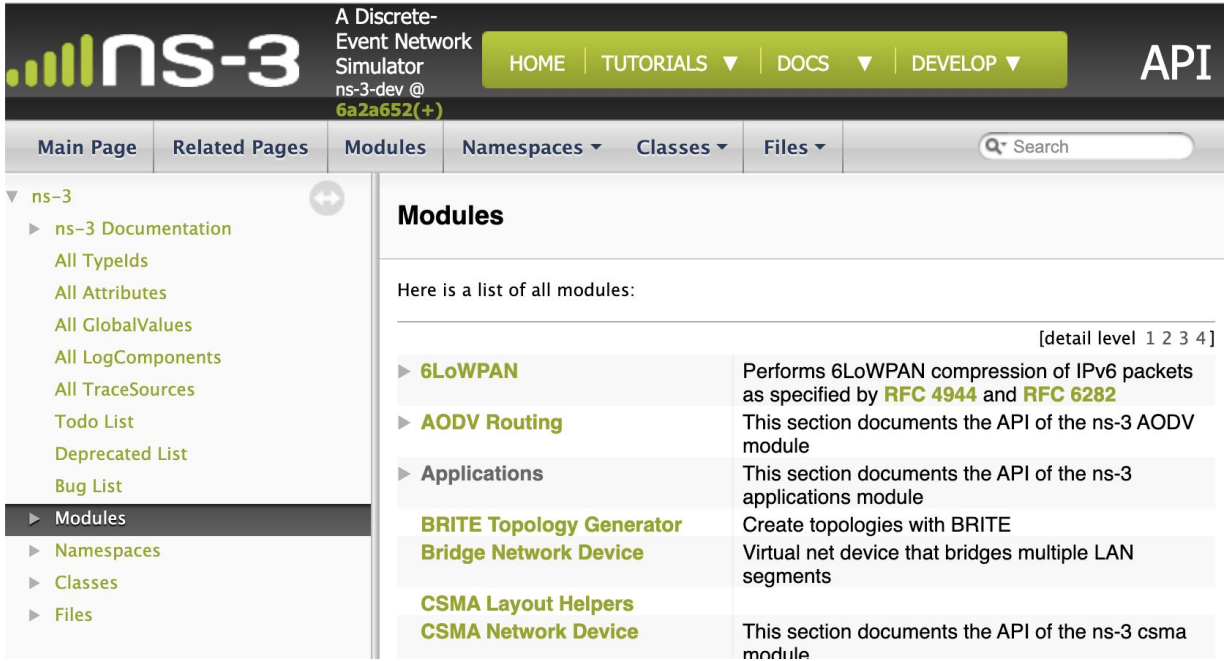
Network Simulator 3

- A discrete-event network simulator for Internet systems, targeted primarily for research and educational use.
  - Uses virtual time to simulate events
- An open source project actively supported by The University of Washington NS-3 Consortium
- Advantages:
  - Inexpensive solution to assess network/protocol performance.
  - Easy monitoring of all sorts of details, ie cwnd
  - Custom topologies can be easily defined to simulate large-scale networks
- Disadvantages:
  - Can not be parallelized
  - May ignore some processing steps/delays

NS-3 Documentation

NS-3 Doxygen:
https://www.nsnam.org/doxygen/

- Search for every module / API / object
- Can also look into the source code



Building NS-3

- `git clone https://github.com/stanford-cs244/ns-3-dev-git.git`

Building NS3

- `./waf configure --build-profile=debug --enable-examples --enable-tests`
- `./waf` (This will take some time to compile)

Testing NS3

- `./test.py` (takes long time)

Running a Script

- `./waf --run hello-simulator`

Detailed Tutorial: https://www.nsnam.org/docs/tutorial/html/

Running Scripts with Arguments

Program Arguments

- `./waf --run '<ns3-program> --arg1=value1 --arg2=value2 ...'`

Debugging

- `./waf --run=hello-simulator --command-template="gdb %s --args <args>"`

Working Directory

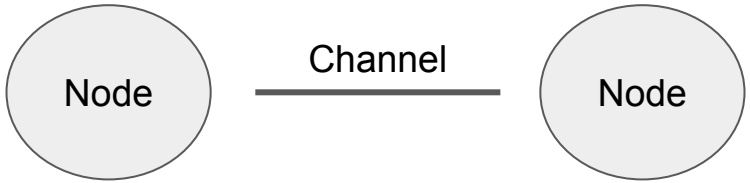
- `./waf --cwd=...`

Key NS-3 Abstractions

- Node (NodeContainer)
  - Basic computing device, end-host
  - Add *applications*, *protocol stacks* and *peripheral cards* with drivers to make it work

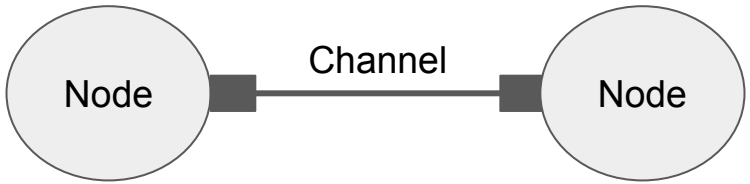
Key NS-3 Abstractions

- Node (NodeContainer)
  - Basic computing device, end-host
  - Add *applications*, *protocol stacks* and *peripheral cards* with drivers to make it work
- Channel (Eg. PointToPointHelper)
  - Basic communication subnetwork
  - Example types: *CsmaChannel*, *PointToPointChannel* and *WifiChannel*



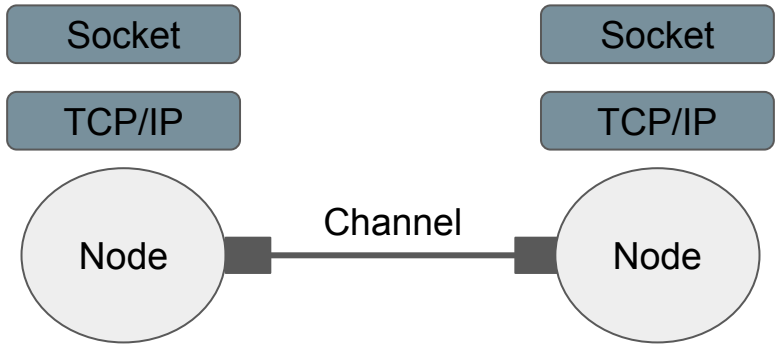
Key NS-3 Abstractions

- Node (NodeContainer)
  - Basic computing device, end-host
  - Add *applications*, *protocol stacks* and *peripheral cards* with drivers to make it work
- Channel (Eg. PointToPointHelper)
  - Basic communication subnetwork
  - Example types: *CsmaChannel*, *PointToPointChannel* and *WifiChannel*
- Net Device (NetDeviceContainer)
  - NIC that is “installed” in a Node
  - Handles L2 networking



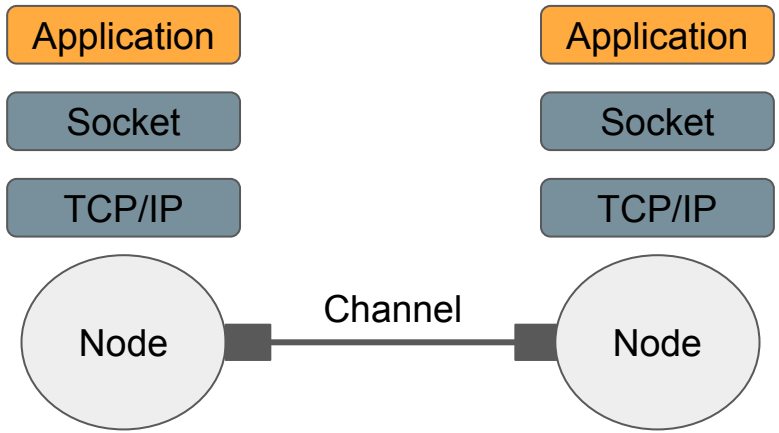
Key NS-3 Abstractions

- Node (NodeContainer)
  - Basic computing device, end-host
  - Add *applications*, *protocol stacks* and *peripheral cards* with drivers to make it work
- Channel (Eg. PointToPointHelper)
  - Basic communication subnetwork
  - Example types: *CsmaChannel*, *PointToPointChannel* and *WifiChannel*
- Net Device (NetDeviceContainer)
  - NIC that is “installed” in a Node
  - Handles L2 networking



Key NS-3 Abstractions

- Node (NodeContainer)
  - Basic computing device, end-host
  - Add *applications*, *protocol stacks* and *peripheral cards* with drivers to make it work
- Channel (Eg. PointToPointHelper)
  - Basic communication subnetwork
  - Example types: *CsmaChannel*, *PointToPointChannel* and *WifiChannel*
- Net Device (NetDeviceContainer)
  - NIC that is “installed” in a Node
  - Handles L2 networking
- Application (ApplicationContainer)
  - Sends and receives packets



Things to Know

- Creating Your Own Model
  - https://www.nsnam.org/docs/manual/html/new-models.html
- Creating Your Own Module
  - https://www.nsnam.org/docs/manual/html/new-modules.html
- NS-3 Way of Logging
  - https://www.nsnam.org/docs/manual/html/logging.html

Starter Code Overview

Grading Rubric

- 3 points: Producing working code that generates the graphs
  - (-1) Some parameters are missing
  - (-2) Some functions are empty
- 5 points: Answering the questions
  - (-1) for each incorrect answer
- 1 point: Producing correct graphs
- 1 point: Design and code quality
  - (-0.5) for each Inconsistent syntax / Hardcoded parameters / Un-commented Non-trivial Complex Logic (max -1)

Good luck!

Final submission due Thursday, April 22 2021, 11:59 pm PT.