Multimedia & Wireless Networking Laboratory

Session 1. ns-3 기초

이재홍
Multimedia & Wireless Networking Laboratory, SNU
jhyi@mwnl.snu.ac.kr

Contents

- Linux Basic Commands
- Introduction to ns-3
- Starting ns-3
- ns-3 Key Abstractions

Linux Basic Commands

Linux Basic Commands

- man [instruction]
  - Manual of the instruction
  - ex) man ls
- ls
  - Short listing of directory contents
  - ex) ls -al
- cd <path>
  - Change directory
  - ex)
    - cd .. (go to the parent directory)
    - cd / (go to the root directory)

Linux Basic Commands

- mkdir <directory name>
  - Make directory
  - mkdir test_dir1
- gedit <filename>
  - Open <filename> with gedit
  - gedit test1.txt
- cat <filename>
  - Print the file to the screen
  - cat test1.txt

Linux Basic Commands

- cp <filename1> <filename2>
  - Copy files or directories
  - ex)
    - cp test1.txt test_dir2/
    - cp –r test_dir1 test_dir2

- mv <filename1> <filename2>
  - Move file
  - Change filename
  - ex)
    - mv test_dir1/test1.txt test_dir2/
    - mv test1.txt test2.txt

Linux Basic Commands

- rm <filename>
  - Remove files and directories
  - ex)
    - rm test2.txt
    - rm –r test_dir2
    - rm *
- grep <pattern> <filename>
  - search for a pattern in a file
  - grep hello test1.txt

Linux Basic Commands

- find <directory> <expressions>
  - search for files in a directory
  - ex)
    - find .
    - find . –name “*.txt”
- find with grep
  - find . –name “*.txt” | xargs grep hello
  - find files which contain ‘hello’

Linux Basic Commands

- tar
  - tar archiving
  - function command
    - c: to create a tar file
    - x: extract the content of the tar file
    - f: specifies the filename
    - v: verbose output
  - ex)
    - tar cvf test.tar test_dir1
    - tar xvf test.tar
    - tar xvf ns-allinone-3.27.tar.gz

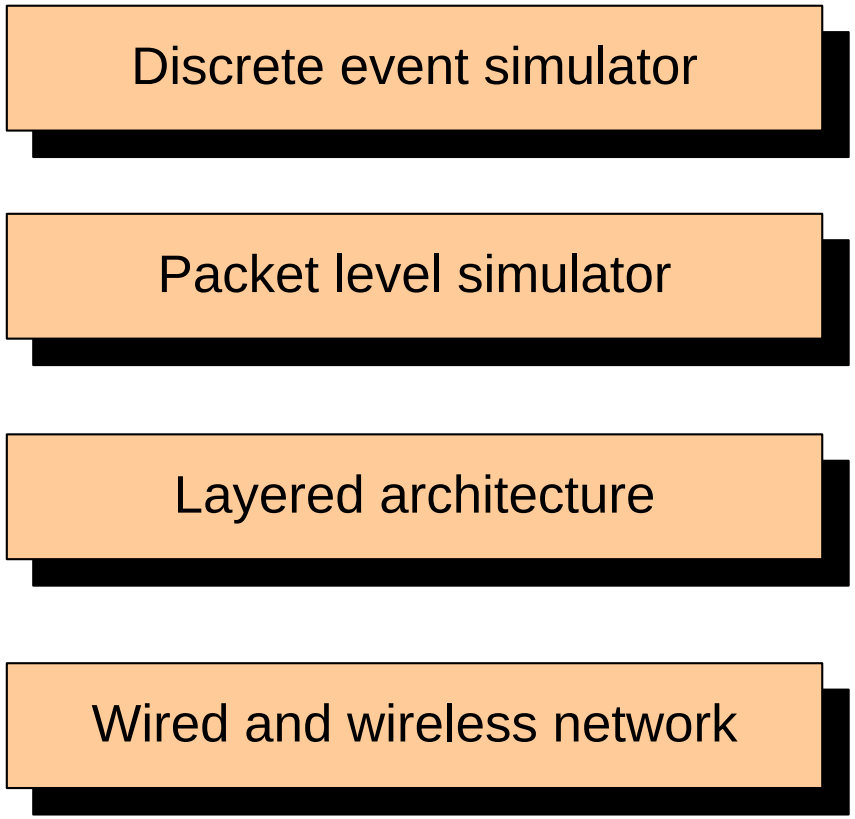
Linux Basic Commands

- Redirection
  - >
    - Save results into a file
    - cat test1.txt > test2.txt
  - >>
    - Append results into a file
    - cat test1.txt >> test2.txt
  - 2>&1
    - stderr → stdout
    - ./waf –-run scratch/scratch-simulator > log.out 2>&1

Introduction to ns-3

ns-3

- ns-3 is targeted at networking research.



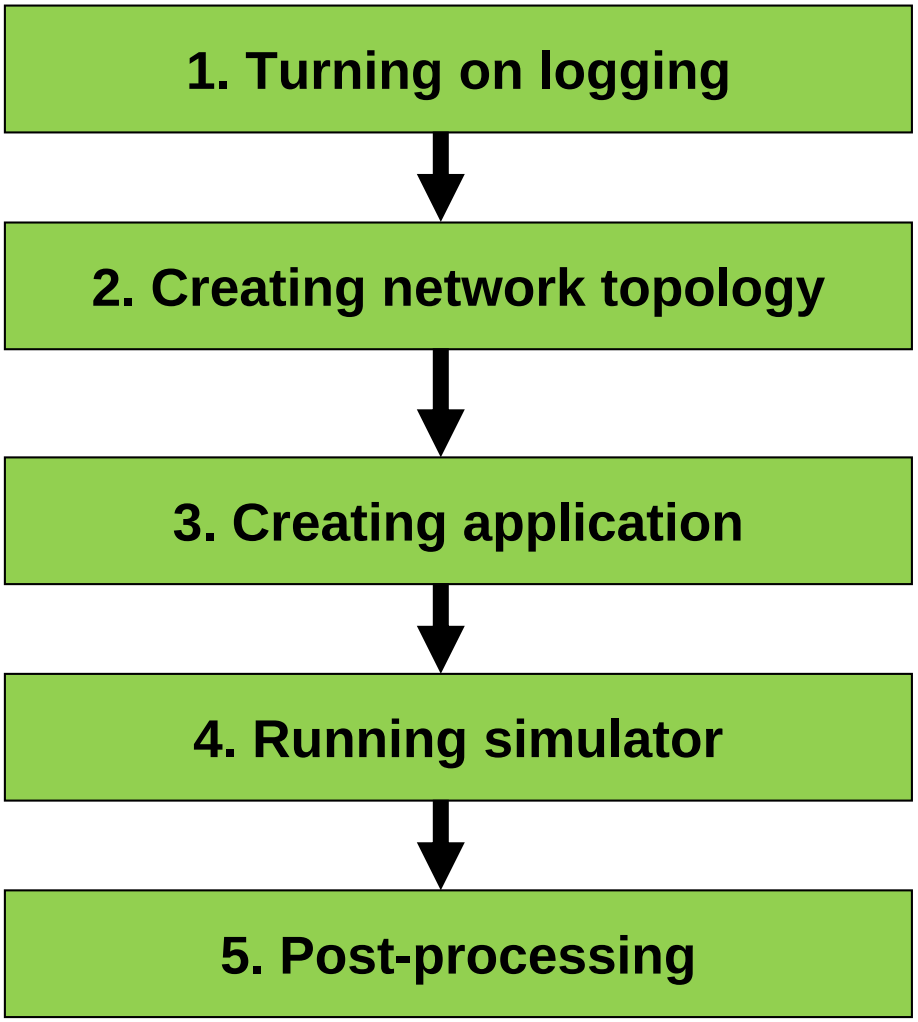
ns-3 Status

- NS-3 is a free open source project
  - Discrete-event network simulator, packet level simulator
  - Layered architecture, wired/wireless network
- Periodic update
  - Current development is **ns-3.29**
- Supporting platform
  - FreeBSD, Linux, SunOS, Solaris, Windows (Cygwin)
- Website
  - *http://www.nsnam.org/*
  - Can find Doxygen, reference manual, and tutorial

Waf Build System

- waf
  - Python-based framework for configuring, compiling, and installing applications
  - Reference
    - http://code.google.com/p/waf
- waf key concepts
  - configure → ./waf –d [optimized|debug] configure
  - make → ./waf
  - make clean → ./waf clean
  - Running program
    - E.g.) ./waf –run simple-point-to-point

ns-3 Simulation Procedure



- Session 2: Logging modules
- Session 3, 4: Tracing system

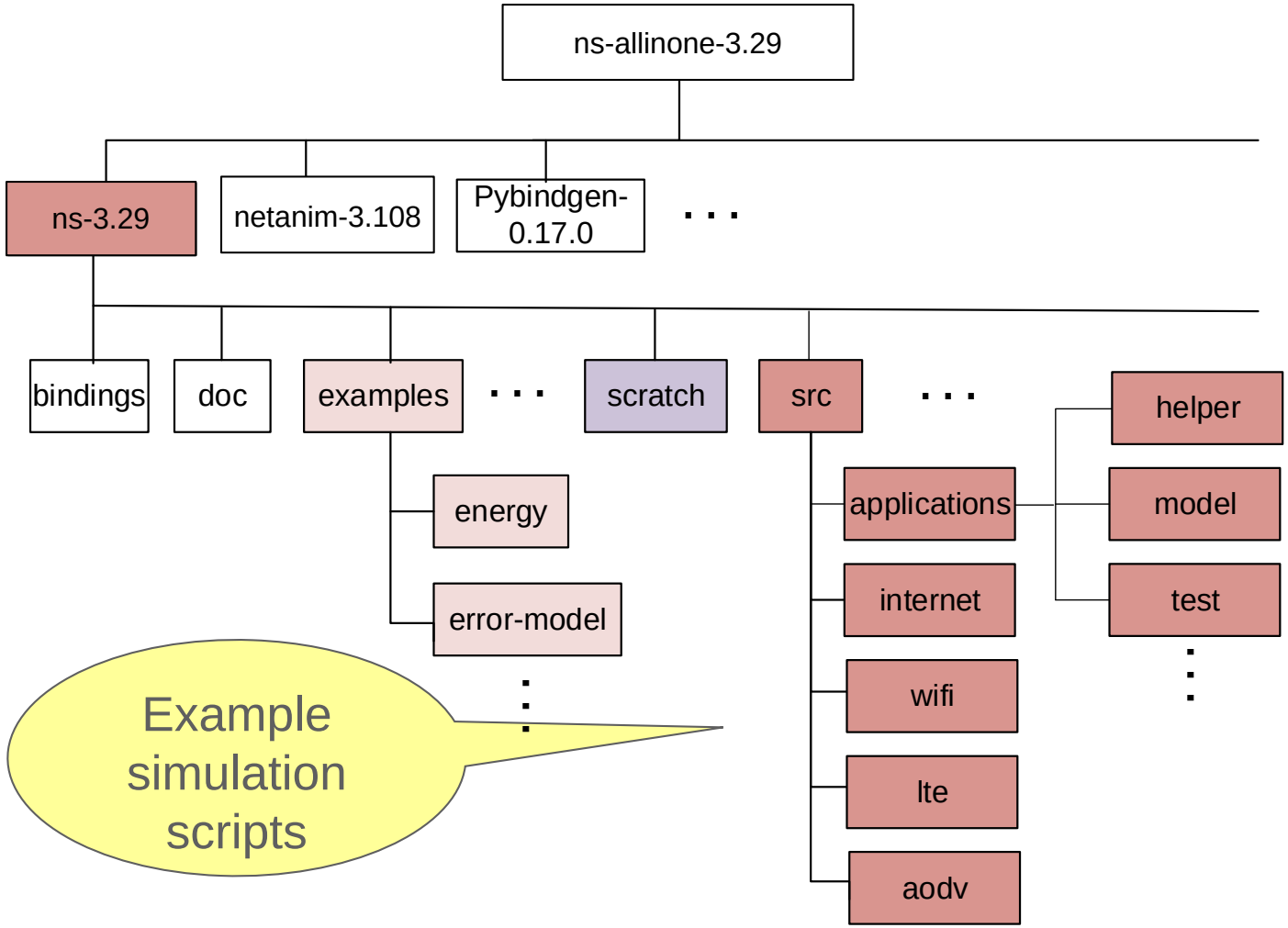
- Session 5, 6, 7
  - Wired LAN, wireless LAN, LTE

- Session 3,4,8
  - Basic applications
  - Create new headers
  - TCP/UDP/IP

- Session 2
  - Command line arguments

- Session 3, 4, 5, 6
  - Wireshark, Gnuplot

ns-3 all-in-one package Directory Structure



Starting ns-3

How to Download ns-3

- ns-3 download: http://www.nsnam.org/ns-3-29/download/
  - [user@com ~]# tar xvf ns-allinone-3.29.tar.bz2
  - [user@com ~]# cd ns-allinone-3.29

- ns-3 download using wget:
  - [user@com ~]# wget https://www.nsnam.org/releases/ns-allinone-3.29.tar.bz2
  - [user@com ~]# tar xvf ns-allinone-3.29.tar.bz2
  - [user@com ~]# cd ns-allinone-3.29

How to Install ns-3

- Building with build.py
  - [user@com ~/ns-allinone-3.29]$ ./build.py
- Building with waf
  - [user@com~/ns-allinone-3.29] $ cd ns-3.29
  - [user@com~/ns-allinone-3.29/ns-3.29] $ ./waf clean
  - Configuration with build-profile=debug for outputting log messages
    - **$ ./waf –-build-profile=debug --enable-examples --enable-tests configure**
  - Configuration with build-profile=optimized for maximizing simulation speed
    - **$ ./waf –-build-profile=optimized --enable-examples --enable-tests configure**
  - Build
    - **$ ./waf**
  - Specifying output directory for build-profile
    - $ ./waf –-build-profile=debug –-out=build/debug
- Test
  - $ ./test.py –c core

Running a Script

- cd ns-allinone-3.29/ns-3.29
- ../ns-3.29$ ./waf --run scratch/scratch-simulator

- Output

```
'build' finished successfully (2.454s)
Scratch Simulator
```

Scratch Simulator

scratch/scratch-simulator.cc

```
#include "ns3/core-module.h"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("ScratchSimulator");

int
main (int argc, char *argv[])
{
  NS_LOG_UNCOND ("Scratch Simulator");

  Simulator::Run ();
  Simulator::Destroy ();
}
```

module inclusion
: automatically includes header files in src/core

namespace declaration
: after that you will not have to type ns3::

definition of logging module name
: will be explained Session 2

Outputting log message
: will be explained Session 2

Running and finishing simulation

Workspace

- ../ns-allinone-3.29/ns-3.29**/scratch**
- Run program only in the scratch folder
- Run program by the commands below
  - ./waf --run scratch/example

ns-3 in Windows

- Basic options for Windows support
  - Virtualization products (VirtualBox, Vmware)
  - How to run simulations using VirtualBox
    - https://www.nsnam.org/wiki/HOWTO_use_VirtualBox_to_run_simulations_on_Windows_machines
  - How to run simulations using VMware
    - http://www.nsnam.org/wiki/HOWTO_use_VMware_to_set_up_virtual_networks_(Windows)
- Cygwin was supported in the past
  - There might be some problems

ns-3 Key Abstractions

Key Abstractions

- 1. Node
  - Host, end system in the Internet
  - Basic computing device abstraction
  - Represented in C++ by the class Node
- 2. Application
  - A user program that generates some activity to be simulated
  - NS-3 applications run on ns-3 Nodes to drive simulations
  - Represented in C++ by the class Application
    - Ex) OnOffApplication, UdpEchoClientApplication
- 3. Channel
  - Medium connected by nodes over which data flows
  - Represented in C++ by the class Channel
    - Ex) CsmaChannel, PointToPointChannel, WifiChannel

Key Abstractions

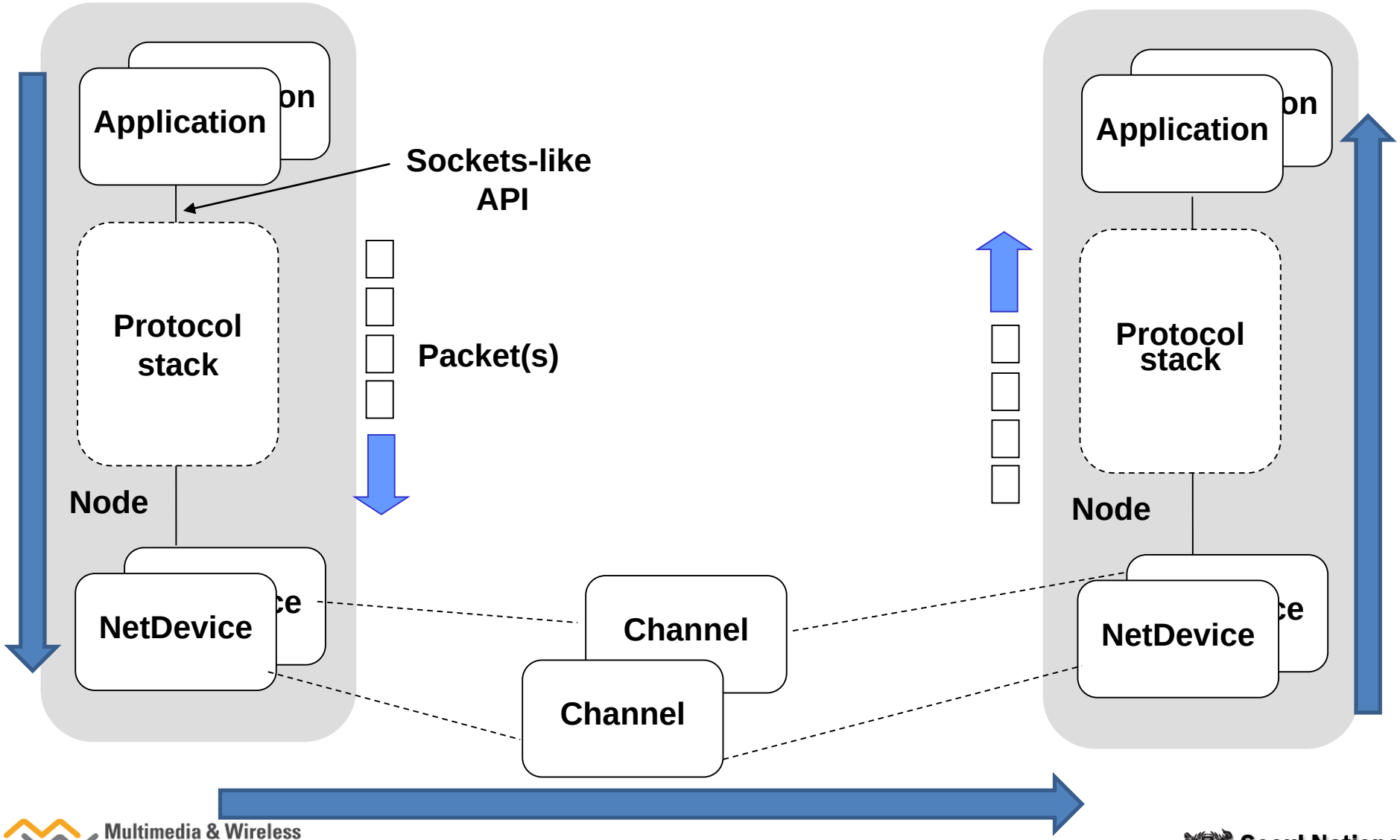
- 4. Net device
  - Like a specific kind of network cable and a hardware device
  - Network Interface Cards; NICs
  - NICs are controlled using the software driver, net devices
  - Represented in C++ by the class NetDevice
  - Provides methods for managing connection to Node and Channel
    - Ex) CsmaNetDevice (work with CsmaChannel)

      WifiNetDevice (work with WifiChannel)
- 5. Topology helpers
  - Topology helpers make ns-3 core operations as easy as possible
  - Create a NetDevice, add an address, install that net device on a Node, configure the node's protocol stack and connect the NetDevice to a Channel

Topology Helpers

- NodeContainer: Provide a convenient way to create, manage and access any Node object
- PointToPointHelper: Configure and connect **PointToPointNetDevice** and **PointToPointChannel** objects. Set DataRate on NetDevices and Delay on Channel
- NetDeviceContainer: To install **NetDevice** at the nodes and create Channel between the nodes
- InternetStackHelper: To install an Internet Stack (TCP, UDP, IP, etc.) on each of the nodes in the node container
- Ipv4AddressHelper: Manage the allocation of IP address and assign addresses to the devices using **Ipv4InterfaceContainer**

NS-3 Basic Simulation Model



Q & A