NS3 Overview

STVN-2016
Jointly organized by
Poornima University, Jaipur & IIIT-Kota

Rahul Hada
hada.rahul@gmail.com
TCP/IP Beginning

The very beginning of TCP/IP
TCP/IP Model

Encapsulation

TCP / UDP

IP

Layer 2

Layer 1

Upper Layers

TCP/UDP Message

Upper Layer Message

TCP/UDP Header

Upper Layer Headers

Upper Layer (Application) Data

Upper Layer Headers

Upper Layer (Application) Data

IP Datagram

Layer 2 Frame

Layer 2 Header

IP Header

TCP/UDP Header

Upper Layer Headers

Upper Layer (Application) Data

Layer 2 Footer

Decapsulation

IP

TCP / UDP

Layer 2

Layer 1

Simulation of TCP Variants using NS-3
Conceptual
Components

- Basic Components
  - Nodes
  - Net Device
  - Channels
  - Application
  - Protocol Stack
Network Elements

- **Nodes** may or may not have mobility
- Nodes have “**Network Devices**”
  - Network Devices *transfer packets over channels*
  - It incorporates **Layer-1 (PHY)** & **Layer-2 (MAC)**
- Devices **interface** with **Layer-3 (Network)**
- **Layer-3 supports** Layer-4 (**Transport**)
- **Layer-4 is used by** **Layer-5**
## NS3: Software Organization

- **Node Class**
  - NetDevice
  - Address types (IPv4, MAC etc.)
  - Queues
  - Sockets

- **Perfrom testing**
  - (contribution of example, core etc.)
  - Test.py

- **High-level wrappers**
  - Aimed at scripting

- **Mobility models**
  - (static, random walk etc)

### Table:

<table>
<thead>
<tr>
<th>test</th>
<th>helper</th>
</tr>
</thead>
<tbody>
<tr>
<td>protocols</td>
<td>applications</td>
</tr>
<tr>
<td>internet</td>
<td>mobility</td>
</tr>
</tbody>
</table>

- **Packets**
  - Packets Header
  - Packets Tags
  - Pcap file writing

- **Logging, Random Variables, Events, Schedulers etc**
Interface: NS3

- Folder structure
  - model/ - contains source code for main part of module
  - helper/ - contains code for helper classes
  - examples/ - contains topology example related to module
  - bindings/ - files related to python
  - wscript – the “Makefile” equivalent
  - doc/ - document API of the module
NS3-Implementation

Node Stack

Application

Socket::Send()

[Tcp/Udp]SocketImpl

::Send()

[Tcp/Udp]L4Protocol

::Send()

Ipv4L3Protocol

::Send()

ArpIpv4Interface

::Lookup()

ArpL3Protocol

::Send()

NetDevice

Simulation of TCP Variants using NS-3
Net Device and Channels

- Net Device are **strongly bound** to Channels of a matching type
- Net Devices examples:
  - *Ethernet NIC*
  - *Wifi Net Device*
- Channel examples:
  - *CSMA Channel*
  - *Wifi Channel*
Net Device and Channel

- **Point-to-point** – PPP link
- **CSMA** – Ethernet link
- **Wifi** – 802.11 link
  - Infrastructure and Ad-hoc
- and, many more
Routing

- Optimized Link State Routing (OLSR)
- Ad hoc On Demand Distance Vector (AODV)
- Destination Sequenced Distance Vector (DSDV)
- Dynamic Source Routing (DSR)
- Ipv4GlobalRouting – used to store routes computed by the global route manager
- and, few more
Application (Traffic Generator)

- **Bulk-Send** – Send data as fast as possible
  - **BulkSendApplication**
- **On-Off** – On off pattern
  - **OnOffApplication**
- **Udp-Server** – Receive UDP packets
  - **UdpServer**, **UdpServerHelper**
- **UDP-Client** – UDP packet with seq no and time stamp
  - **UdpClient**, **UdpClientHelper**
- **V4ping** – Sends one ICMP ECHO request, report the RTT
  - **ping6**
  - **V4ping**, **V4pingHelper**
Helper Classes

- It provides a **set of classes** and **methods** that make common operations easier than using the low-level API

- **Absence of Helper**: Simulation program will be quite long and tedious

- Contains:
  - Container classes
  - Helper classes

- It is implemented using the low-level API
Main Program Structure

- Include HEADER files
- Include NAMESPACE
- Enable /disable LOGGING
- Create NODE
- Configure TOPOLOGY HELPER for Nodes
- Set up INTERNET STACK
- Set up APPLICATION
- Run SIMULATION
Example Script-1

```cpp
/* -*- Mode:C++; c-file-style:"gnu"; indent-tabs-mode:nil; -*- */
// GPLv2 Licence ...

#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/applications-module.h"
using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("FirstScriptExample");

int main (int argc, char *argv[]) {
  LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
  LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);

  NodeContainer nodes;
  nodes.Create (2);

  PointToPointHelper pointToPoint;
  pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps");
  pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms");

  NetDeviceContainer devices;
  devices = pointToPoint.Install (nodes);
```
Example Script-1

```cpp
InternetStackHelper stack;
stack.Install (nodes);

Ipv4AddressHelper address;
address.SetBase("10.1.1.0", "255.255.255.0");

Ipv4InterfaceContainer interfaces = address.Assign (devices);

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);
echoClient.SetAttribute("MaxPackets", UintegerValue (1));
echoClient.SetAttribute("Interval", TimeValue (Seconds (1.0)));
echoClient.SetAttribute("PacketSize", UintegerValue (1024));

ApplicationContainer clientApps = echoClient.Install (nodes.Get (0));
clientApps.Start (Seconds (2.0));
clientApps.Stop (Seconds (10.0));

Simulator::Run ();
Simulator::Destroy ();
return 0;
```
Running Example

- Create.
- Modify-Copy programs in scratch
- ./waf -run
  /scratch/your_scriptname
What is the use of Helper Class?
What are the types of application traffic generators?
What is NODE?
Why we use Net Device?
What is the use of Channel?
To be Continued...