Multimedia & Wireless Networking Laboratory

Session 5. Wired LAN in ns-3

황선욱

Multimedia & Wireless Networking Laboratory, SNU

swhwang@mwnl.snu.ac.kr

Contents

- Multiple Access Protocols
- CSMA Model in ns-3
- Bus Topology Example
- Bridge Model in ns-3
- Star Topology Example

Multiple Access Protocols

Multiple Access Protocols

- Single shared broadcast channel
- Two or more simultaneous transmissions by nodes
  - Collision if node receives two or more signals at the same time

Multiple Access Protocol

- Protocol for how nodes share channel, i.e., determining when node can transmit
- In general, no extra control channel
  - No out-of-band channel for coordination
- In many cases, no central coordinator
  - Distributed protocol required

Random Access Protocols

- When node has packet to send
  - Transmit at full channel data rate *R*
  - No *a priori* coordination among nodes
- Two or more transmitting nodes ➜ collision
- Random access MAC protocol specifies
  - How to detect collisions & recover from collisions (e.g., via delayed retransmissions)
- Examples of random access MAC protocols
  - Slotted ALOHA, ALOHA
  - CSMA/CD, CSMA/CA

CSMA (Carrier Sense Multiple Access)

- **CSMA**: Listen before transmit
  - If channel sensed idle, transmit entire frame
  - If channel sensed busy, defer transmission
  - Human analogy: Don't interrupt others!

- **CSMA/CD (Collision Detection)**
  - Collisions *detected* within short time
  - Colliding transmissions aborted, reducing channel wastage
  - Human analogy: Polite conversationalist
  - Collision detection
    - Easy in wired LANs: Measure signal strengths, compare transmitted & received signals
    - Difficult in wireless LANs: Receiver shut off while transmitting

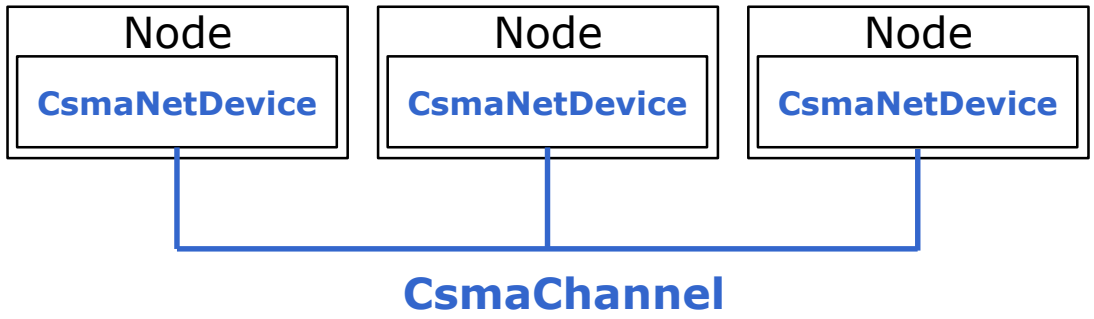
CSMA Model in ns-3

MAC Model in ns-3

- NetDevice class
  - AlohaNoAckNetDevice
  - BridgeNetDevice
  - CsmaNetDevice
  - FdNetDevice
  - LoopbackNetDevice
  - LrWpanNetDevice
  - LteNetDevice
  - NonCommunicatingNetDevice
  - OpenFlowSwitchNetDevice
  - PointToPointNetDevice
  - SimpleNetDevice
  - UanNetDevice
  - VirtualNetDevice
  - WaveNetDevice
  - WifiNetDevice
  - WimaxNetDevice

CSMA Model in ns-3

- Model a simple bus network in the spirit of Ethernet
- Not model collision detection
- Related classes & helper
  - CsmaChannel
  - CsmaNetDevice
  - CsmaHelper



CsmaChannel

- Not consider the distances between stations or the speed of light to determine collisions
- No limit for the number of devices connected to the channel
- Attributes
  - **DataRate**: Transmission data rate of devices connected to the channel
    - Initial value: 4294967295 bps
  - **Delay**: Transmission delay through the channel
    - Initial value: +0.0 ns
- States
  - IDLE, TRANSMITTING, and PROPAGATING
  - IDLE → TRANSMITTING → PROPAGATING → IDLE

CsmaNetDevice

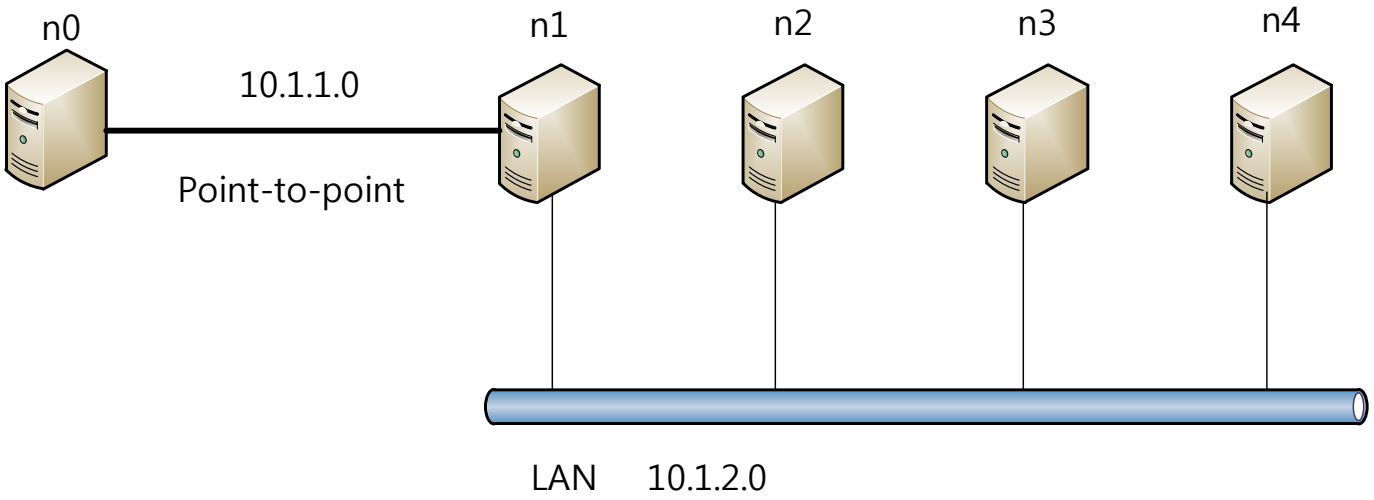
- Attributes
  - **Address**: The MAC address of this device
    - Initial value: ff:ff:ff:ff:ff:ff
  - **Mtu**: The MAC-level Maximum Transmission Unit in byte
    - Initial value: 1500
  - **SendEnable**: Enable or disable the transmitter section of the device
  - **ReceiveEnable**: Enable or disable the receiver section of the device
  - **ReceiveErrorModel**: Receiver error model used to simulate packet loss
  - **TxQueue**: A queue to use as the transmit queue in the device.

- TraceSources
  - MacTx
  - MacTxDrop
  - MacPromiscRx
  - PhyTxEnd
  - PhyTxDrop
  - PhyRxEnd
  - PhyRxDrop
  - Sniffer
  - PromiscSniffer

Bus Topology Example

Topology

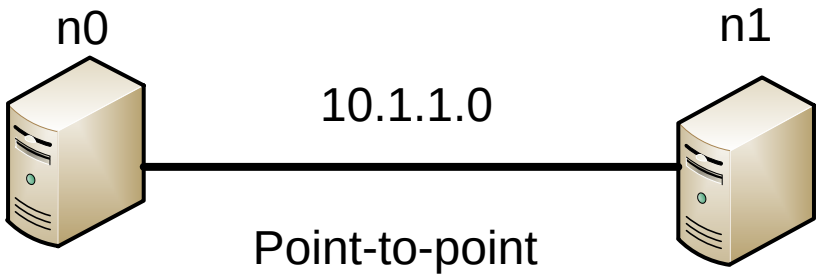
- CSMA Network topology



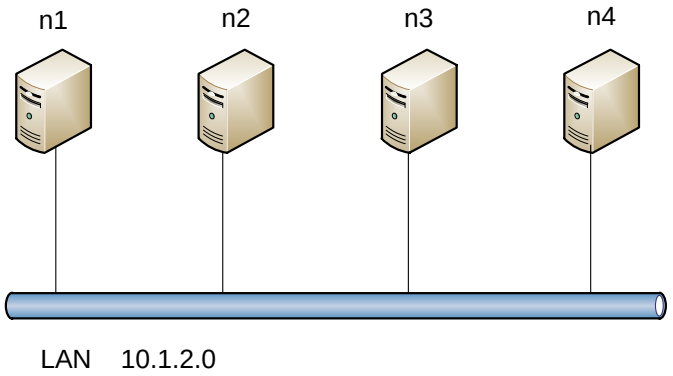
- UDP Echo from n0 to n4

CSMA Node Creation

- P2P Node creation
  - NodeContainer p2pNodes;
  - p2pNodes.Create (2);



- Csma Node creation
  - NodeContainer csmaNodes;
  - csmaNodes.Add (p2pNodes.Get (1));
  - csmaNodes.Create (nCsma);



CsmaHelper

- Build a set of CsmaNetDevice objects
- CSMA Channel attributes modification
  - CsmaHelper csma;
  - csma.SetChannelAttribute ("DataRate", StringValue ("100Mbps"));
  - csma.SetChannelAttribute ("Delay", TimeValue (NanoSeconds (6560)));
- CSMA Net Device creation
  - NetDeviceContainer csmaDevices;
  - csmaDevices = csma.Install (csmaNodes);
  - csmaDevices = csma.Install (NodeContainer (n0, n1));

CSMA Example Code (1/5)

```
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/csma-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/applications-module.h"
#include "ns3/ipv4-global-routing-helper-module.h"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("s5_ex1");

int
main (int argc, char *argv[])
{
        bool verbose = true;
        uint32_t nCsma = 3;
```

To determine whether the logging components are enabled or not

CSMA Example Code (2/5)

```
commandLine cmd;
cmd.AddValue ("nCsma", "Number of \"extra\" CSMA nodes/devices", nCsma);
cmd.AddValue ("verbose", "Tell echo applications to log if true", verbose);
cmd.Parse (argc,argv);

if (verbose)
{
      LogComponentEnable("UdpEchoClientApplication", LOG_LEVEL_INFO);
      LogComponentEnable("UdpEchoServerApplication", LOG_LEVEL_INFO);
}

nCsma = nCsma == 0 ? 1 : nCsma;

NodeContainer p2pNodes;
p2pNodes.Create (2);

NodeContainer csmaNodes;
csmaNodes.Add (p2pNodes.Get (1));
csmaNodes.Create (nCsma);
```

Separated Node generation for p2p and csma

Node container for csma includes p2p second node

CSMA Example Code (3/5)

```
PointToPointHelper pointToPoint;
pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));
NetDeviceContainer p2pDevices;
p2pDevices = pointToPoint.Install (p2pNodes);

/* CSMA link configuration */
CsmaHelper csma;
csma.SetChannelAttribute ("DataRate", StringValue ("100Mbps"));
csma.SetChannelAttribute ("Delay", TimeValue (NanoSeconds (6560)));

NetDeviceContainer csmaDevices;
csmaDevices = csma.Install (csmaNodes);

/* Install Internet stack and assign IP addresses */
InternetStackHelper stack;
stack.Install (p2pNodes.Get (0));
stack.Install (csmaNodes);
```

CSMA Example Code (4/5)

```
Ipv4AddressHelper address;

address.SetBase ("10.1.1.0", "255.255.255.0");
Ipv4InterfaceContainer p2pInterfaces;
p2pInterfaces = address.Assign (p2pDevices);

address.SetBase ("10.1.2.0", "255.255.255.0");
Ipv4InterfaceContainer csmaInterfaces;
csmaInterfaces = address.Assign (csmaDevices);

UdpEchoServerHelper echoServer (9);
ApplicationContainer serverApps = echoServer.Install (csmaNodes.Get (nCsma));
serverApps.Start (Seconds (1.0));
serverApps.Stop (Seconds (10.0));

UdpEchoClientHelper echoClient (csmaInterfaces.GetAddress (nCsma), 9);
echoClient.SetAttribute ("MaxPackets", UintegerValue (1));
echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.)));
echoClient.SetAttribute ("PacketSize", UintegerValue (1024));
```

CSMA Example Code (5/5)

```
  ApplicationContainer clientApps = echoClient.Install (p2pNodes.Get (0));
  clientApps.Start (Seconds (2.0));
  clientApps.Stop (Seconds (10.0));

  Ipv4GlobalRoutingHelper::PopulateRoutingTables ();

  pointToPoint.EnablePcapAll ("s5_ex1");
  csma.EnablePcap ("s5_ex1", csmaDevices.Get (1), true);

  Simulator::Run ();
  Simulator::Destroy ();
  return 0;
}
```

Bridge Model in ns-3

Bridge Model in ns3

- BridgeNetDevice is a virtual net device that bridges multiple LAN segments

- Implementation
  - By default a "learning bridge" algorithm is implemented
    - Incoming unicast frames from one port may occasionally be forwarded throughout all other ports
    - But usually they are forwarded only to a single correct output port.
  - Bridging is designed to work only with NetDevices modelling IEEE 802-style technologies, such as CsmaNetDevice and WifiNetDevice.
  - WifiNetDevice in adhoc mode is not supported with bridging.

- Related classes & helper
  - BridgeChannel
  - BridgeNetDevice
  - BridgeHelper

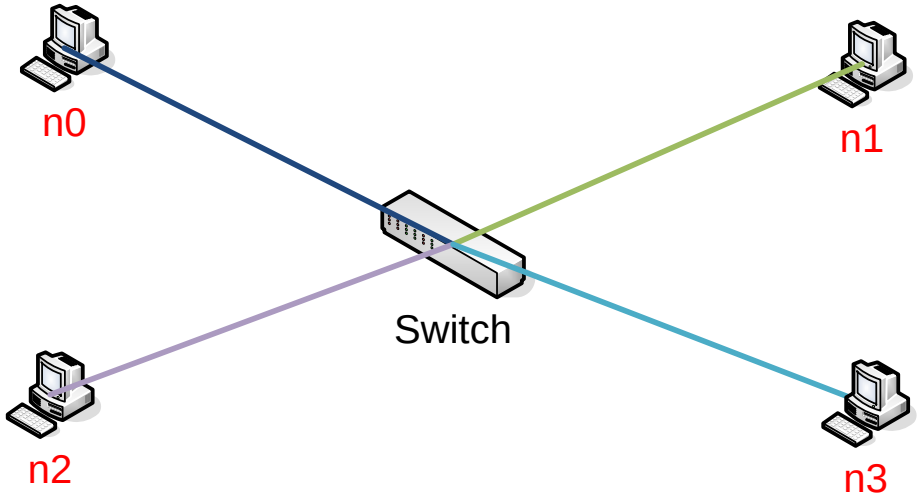
BridgeNetDevice

- Attributes
  - **Mtu**: The MAC-level Maximum Transmission Unit in byte
    - Initial value: 1500
  - **EnableLearning**: Enable the learning mode of the Learning Bridge
    - Initial value: true
  - **ExpirationTime**: Time it takes for learned MAC state entry to expire
    - Initial value: +300000000000.0ns

- No TraceSources

Star Topology Example

Topology



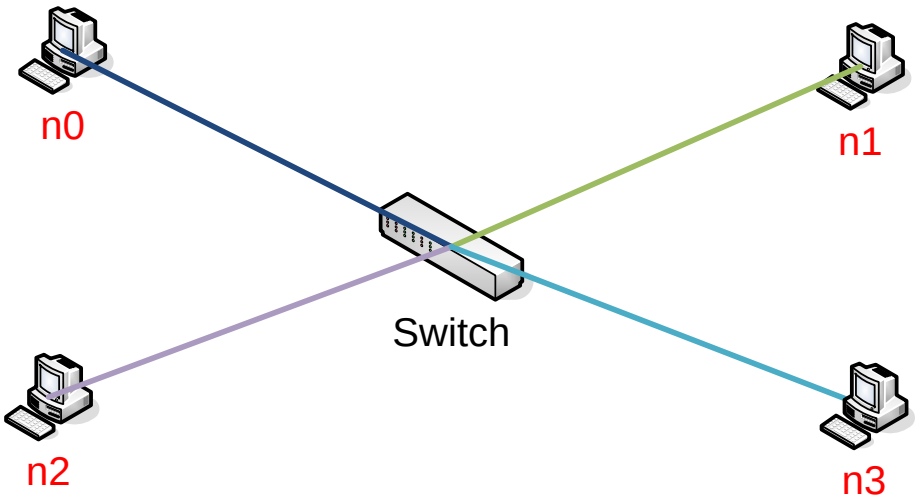
- Csma link: DataRate 5 Mbps, Delay 10 us
- Flows
  - Onoff Application
  - Flow 1: n0 → n1, UDP 5 Mbps, 1−10 s
    - OnTime= OffTime= 1 s
  - Flow 2: n3 → n0, UDP 10 Mbps, 3−13s
    - OnTime= 0.3 s, OffTime= 0.7 s

Node Creation and CsmaHelper

- NodeContainer for terminal nodes
  - NodeContainer terminals;
  - terminals.Create (4);
- NodeContainer for the SwitchNode
  - NodeContainer csmaSwitch;
  - csmaSwitch.Create (1);
- CsmaHelper
  - CsmaHelper csma;
  - csma.SetChannelAttribute ("DataRate", DataRateValue (5000000));
  - csma.SetChannelAttribute ("Delay", TimeValue (MilliSeconds (2)));

CSMA Link Creation

- CSMA links from each terminal to the switch
  - NetDeviceContainer terminalDevices;
  - NetDeviceContainer switchDevices;
  - for (int i = 0; i < 4; i++)
    {
    NetDeviceContainer link = csma.Install (NodeContainer (terminals.Get (i), csmaSwitch));
    terminalDevices.Add (link.Get (0));
    switchDevices.Add (link.Get (1));
    }



BridgeHelper

- Add capability to bridge multiple LAN segment
- Bridge NetDevice creation
  - Ptr<Node> switchNode = csmaSwitch.Get (0);
  - BridgeHelper bridge;
  - bridge.Install (switchNode, switchDevices);
- BridgeHelper class function

NetDeviceContainer Install (Ptr< Node > node, NetDeviceContainer c)

Example Code (1/6)

```
#include <iostream>
#include <fstream>
…

/* Trace sink using context */
static void
Rxtime (std::string context, Ptr<const Packet> p, const Address &a)
{
    static double bytes1, bytes2=0;
    if (context == "Flow1"){
        bytes1+=p->GetSize();
        NS_LOG_UNCOND("1\t" << Simulator::Now().GetSeconds()
            << "\t" << bytes1*8/1000000/(Simulator::Now().GetSeconds()-1));
    } else if (context == "Flow2"){
        bytes2+=p->GetSize();
        NS_LOG_UNCOND("2\t" << Simulator::Now().GetSeconds()
            << "\t" << bytes2*8/1000000/(Simulator::Now().GetSeconds()-3));
    }
}
```

Example Code (2/6)

```
int
main (int argc, char *argv[]) {
      CommandLine cmd;
      cmd.Parse (argc, argv);

      /* Create nodes */
      NodeContainer terminals;
      terminals.Create (4);

      NodeContainer csmaSwitch;
      csmaSwitch.Create (1);

      /* Create links */
      CsmaHelper csma;
      csma.SetChannelAttribute ("DataRate", DataRateValue (5000000));
      csma.SetChannelAttribute ("Delay", TimeValue (MicroSeconds (10)));

      NetDeviceContainer terminalDevices;
      NetDeviceContainer switchDevices;
```

Example Code (3/6)

```
/* Connect switch node and terminal node */
for (int i = 0; i < 4; i++)
{
    NetDeviceContainer link = csma.Install (NodeContainer (terminals.Get(i),
        csmaSwitch));
    terminalDevices.Add (link.Get (0));
    switchDevices.Add (link.Get (1));
}

Ptr<Node> switchNode = csmaSwitch.Get (0);
BridgeHelper bridge;
bridge.Install (switchNode, switchDevices);

/* Install Internet stack and assign IP addresses */
InternetStackHelper internet;
internet.Install (terminals);
Ipv4AddressHelper ipv4;
ipv4.SetBase ("10.1.1.0", "255.255.255.0");
ipv4.Assign (terminalDevices);
```

Example Code (4/6)

```
/* Implement OnOff application */
uint16_t port = 9;
OnOffHelper onoff ("ns3::UdpSocketFactory",
    Address (InetSocketAddress (Ipv4Address ("10.1.1.2"), port)));
onoff.SetAttribute ("OnTime",
    StringValue("ns3::ConstantRandomVariable[Constant=1.0]") );
onoff.SetAttribute ("OffTime",
    StringValue("ns3::ConstantRandomVariable[Constant=1.0]") );
onoff.SetAttribute ("DataRate", DataRateValue(5000000));

ApplicationContainer app = onoff.Install (terminals.Get (0));
app.Start (Seconds (1.0));
app.Stop (Seconds (10.0));
```

Example Code (5/6)

```
/* Implement packet sink application to receive OnOff application traffic*/
PacketSinkHelper sink ("ns3::UdpSocketFactory",
     Address (InetSocketAddress (Ipv4Address::GetAny (), port)));
 ApplicationContainer sinkApp1 = sink.Install (terminals.Get (1));
 sinkApp1.Start (Seconds (1.0));
 sinkApp1.Get(0)->TraceConnect ("Rx", "Flow1", MakeCallback (&Rxtime));
```

Connect Rx trace source with *Context*

```
/* Create a similar flow from n3 to n0, starting at time 3 seconds */
onoff.SetAttribute ("Remote",
     AddressValue (InetSocketAddress (Ipv4Address ("10.1.1.1"), port)));
onoff.SetAttribute ("DataRate", DataRateValue(10000000));
onoff.SetAttribute ("OnTime",
     StringValue("ns3::ConstantRandomVariable[Constant=0.3]") );
onoff.SetAttribute ("OffTime",
     StringValue("ns3::ConstantRandomVariable[Constant=0.7]") );
```

Example Code (6/6)

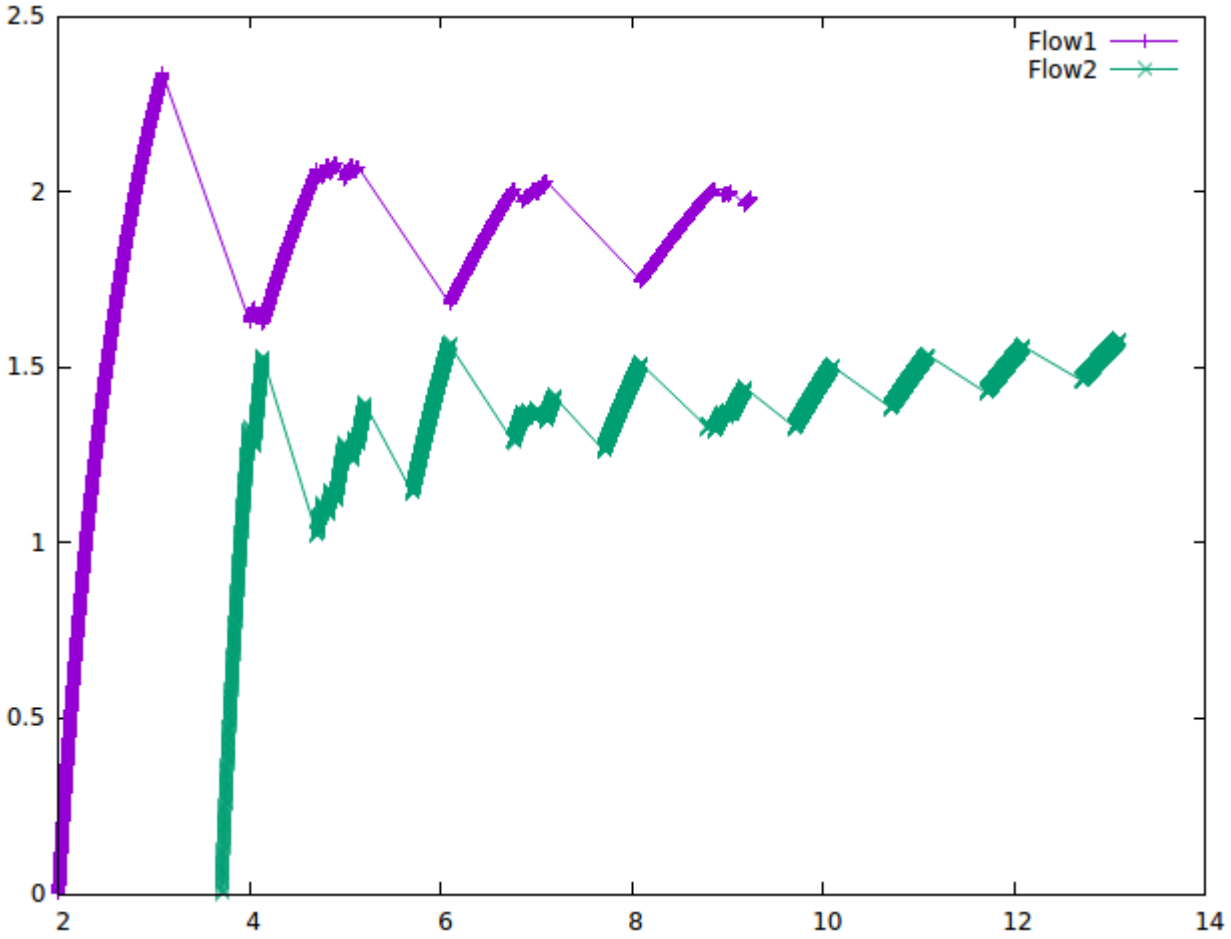
```
    ApplicationContainer app2 = onoff.Install (terminals.Get (3));
    app2.Start (Seconds (3.0));
    app2.Stop (Seconds (13.0));

    ApplicationContainer sinkApp2 = sink.Install (terminals.Get (1));
    sinkApp2.Start (Seconds (1.0));
    sinkApp2.Get(0)->TraceConnect ("Rx", "Flow2", MakeCallback (&Rxtime));

    /* Enable Pcap tracing */
    csma.EnablePcapAll ("s5_ex2", false);

    Simulator::Stop(Seconds (15));
    Simulator::Run ();
    Simulator::Destroy ();
}
```

Connect Rx trace source with *Context*

Example Result (1/2)

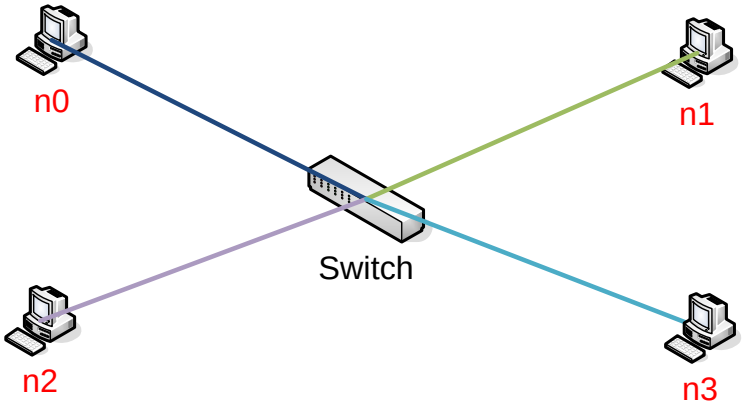
- Save output to a file by redirection
  - ns-3.26$ ./waf --run "s5_ex2" >& log.dat
- Process the output file using a simple *Awk* command
  - awk '$1=="1" {print $2 "\t" $3}' log.dat >& flow1.dat
  - awk '$1=="2" {print $2 "\t" $3}' log.dat >& flow2.dat
- Plot using *Gnuplot*
  - gnuplot
  - plot "flow1.dat" using 1:2 title 'Flow1' with linespoints,
    "flow2.dat" using 1:2 title 'Flow2' with linespoints
  - exit

Example Result (2/2)



In-class Assignment

- Using Switch Example code, make a complete code based on following topology



  - Csma link: DataRate 5 Mbps, Delay 10 us
  - Flows
    - OnOff Application
    - Flow 1: n0->n1, UDP 5 Mbps, 1 – 10 s (always on)
    - Flow 2: n3->n0, UDP 10 Mbps, 3 – 13 s (always on)
  - Draw(trace) throughput (moving average with window size 100) curves regarding both flows

In-class Assignment Result