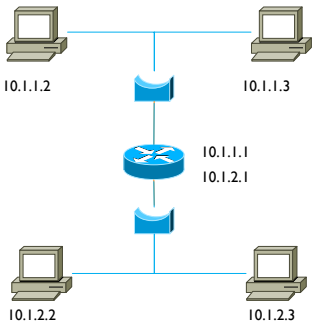
Communication Systems

CCE 3301



Using Pointers to Create the required Nodes

- Ptr<Node> n0 = CreateObject<Node> ();
- Ptr<Node> n1 = CreateObject<Node> ();
- Ptr<Node> n2 = CreateObject<Node> ();
- Ptr<Node> n3 = CreateObject<Node> ();
- Ptr<Node> n4 = CreateObject<Node> ();

- Ptr<Node> bridge1 = CreateObject<Node> ();
- Ptr<Node> bridge2 = CreateObject<Node> ();

n0 and n1 – LAN1
n3 and n4 – LAN2
n2 - router

Wired Networking

Use the **CsmaHelper** – to create **CsmaNetDevice** objects so that we can use a Csma Network Link – similar to using an Ethernet cable to create a wired network

- CsmaHelper csma;
- csma.SetChannelAttribute ("DataRate", DataRateValue (5000000));
- csma.SetChannelAttribute ("Delay", TimeValue (MilliSeconds (2)));

For more information on the class **CsmaHelper**:

- http://www.nsnam.org/docs/release/3.11/doxygen/classns3_1_1_csma_helper.html

NetDeviceContainer

Required NetDeviceContainer:

one for the two nodes and the router representing LAN1

one for the two nodes and the router representing LAN2

Two other NetDeviceContainer for bridge1's connections with the nodes in LAN1 and for bridge2's connections with the nodes in LAN2

- NetDeviceContainer topLanDevices;
- NetDeviceContainer topBridgeDevices;
- NetDeviceContainer bottomLanDevices;
- NetDeviceContainer bottomBridgeDevices;

Create the necessary NodeContainers and NetDeviceContainer

Create a NodeContainer called topLan using 3 pointers

- NodeContainer topLan (n2, n0, n1);

- for (int i = 0; i < 3; i++)
- {

Create a temporary NodeContainer consisting of one of the nodes in topLan and the bridge
Creates a csma Channel between the two nodes and a CsmaNetDevice is installed in each device

- NetDeviceContainer link = csma.Install (NodeContainer (topLan.Get (i), bridge1));

Separating the two CsmaNetDevices created using the previous command

- topLanDevices.Add (link.Get (0));
- topBridgeDevices.Add (link.Get (1));
- }

Create the bridge netdevice, which will do the packet switching. The bridge lives on the node bridge1 and bridges together the topBridgeDevices which are the three CSMA net devices on the node in the diagram above.

- BridgeHelper bridge;
- bridge.Install (bridge1, topBridgeDevices);

Repeat for the other LAN

InternetStackHelper

aggregate IP/TCP/UDP functionality to all the nodes except the bridges

- NodeContainer routerNodes (n0, n1, n2, n3, n4);
- InternetStackHelper internet;
- internet.Install (routerNodes);

For more information regarding the InternetStackHelper:

http://www.nsnam.org/docs/release/3.11/doxygen/classns3_1_1_internet_stack_helper.html

Assign IP addresses
Populate the Routing Tables

- Ipv4AddressHelper ipv4;
- ipv4.SetBase ("10.1.1.0", "255.255.255.0");
- ipv4.Assign (topLanDevices);
- ipv4.SetBase ("10.1.2.0", "255.255.255.0");
- ipv4.Assign (bottomLanDevices);

Notice that the node acting as the router will have two IP addresses

The Bridges will not have any IP addresses

The nodes which we have chosen as router nodes will set up routing tables

- Ipv4GlobalRoutingHelper::PopulateRoutingTables ();

Without this command you cannot transmit packets from one LAN to another

OnOffApplication

Using the OnOffHelper
Generates streams, alternating on-and-off periods
Can be configured to generate many types of traffic
E.g. OnTime=1 and OffTime=0 means CBR
Works with either UDP or TCP – by defining a different protocol

ns3::OnOffHelper::OnOffHelper (std::string protocol, Address address)
Address - the address of the remote node to send traffic to

```
OnOffHelper onoff ("ns3::UdpSocketFactory",
                   Address (InetSocketAddress (Ipv4Address ("10.1.1.3"), port)));
onoff.SetAttribute ("OnTime", RandomVariableValue (ConstantVariable (1)));
onoff.SetAttribute ("OffTime", RandomVariableValue (ConstantVariable (0)));
```

For more details regarding the Attributes you can define for the OnOffApplication check:
http://www.nsnam.org/docs/release/3.11/doxygen/classns3_1_1_on_off_application.html and the GetTypeId section

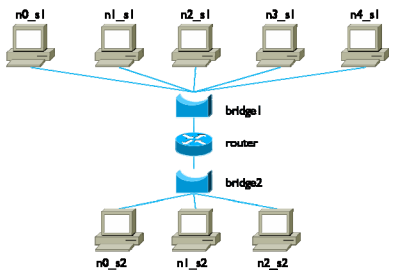
Choose node n0 as the source

```
ApplicationContainer app = onoff.Install (n0);
// Start the application
app.Start (Seconds (1.0));
app.Stop (Seconds (10.0));
```

Sending data across LANs

```
onoff.SetAttribute ("Remote",
                    AddressValue (InetSocketAddress (Ipv4Address ("10.1.1.2"), port)));
ApplicationContainer app2 = onoff.Install (n3);
app2.Start (Seconds (1.1));
app2.Stop (Seconds (10.0));
```

Use the same OnOffHelper object as you did before but change its Attribute using the .SetAttribute and install it in a different node



Assignment

Create the following network

Create a CBR stream:
from n0_S1 to n4_s1
from n1_S1 to n3_s1
from n0_S2 to n2_s2
from n1_S2 to n2_s1

Use only **PcapTracing**