Introduction to NS-3

COMPUTER NETWORKING

ALEX DAWSON

What is NS-3?

- Discrete event network simulator

- Open Source

- Collection of C++ libraries, not an program

- Supported under Linux, FreeBSD and Cygwin

Installing NS-3

Recommended install workflow (alternatives shown in tutorial)

```
mkdir workspace
cd workspace
wget http://www.nsnam.org/release/ns-allinone-3.18.tar.bz2
tar xjf ns-allinone-3.18.tar.bz2
cd ns-allinone-3.18/
./build.py
```

Tutorial : http://www.nsnam.org/docs/release/3.18/tutorial/ns-3-tutorial.pdf

Key Terms and Abstractions

- Node - the hardware (eg. Router, PC, Phone)

- Network Device - transmits and receives over the channel

- Channel – transmission medium between NDs (eg. Wifi, Ethernet)

- Application – creates or receives data sent between nodes

- Helper – NS-3 construct used to quickly configure and create the above

Node		Node
Application		Application
Device	Channel →	Device
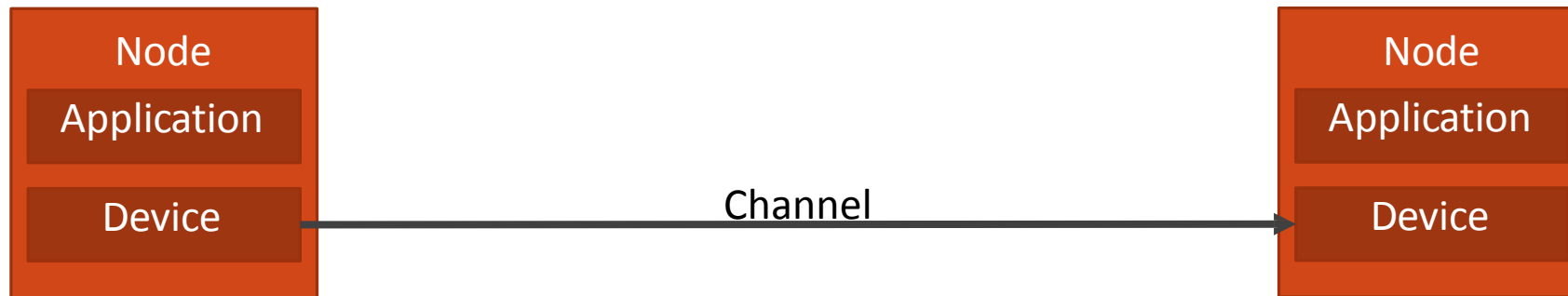
Walkthrough of Example Script

Sections of the code to cover :

- Node Creation

- Channel Configuration

- WiFi Settings

- Mobility and positioning

- Application Configuration

- Running the Simulation

- Flow Monitor

Walkthrough of Example Script (cont)

```
NodeContainer wifiStaNodes;
wifiStaNodes.Create (1);
NodeContainer wifiApNode;
wifiApNode.Create(1);
```

- Node Container
  - Holds groups of nodes
  - Offer functions for node creation and adding existing nodes or containers to the group
- Separate containers for AP and Stations for ease of application/device installation

Walkthrough of Example Script (cont)

```cpp
YansWifiChannelHelper channel;
channel.SetPropagationDelay ("ns3::ConstantSpeedPropagationDelayModel");
channel.AddPropagationLoss("ns3::LogDistancePropagationLossModel");
if(rayleigh){
  wifiChannel.AddPropagationLoss("ns3::NakagamiPropagationLossModel",
                                 "m0", DoubleValue(1.0),
                                 "m1", DoubleValue(1.0),
                                 "m2", DoubleValue(1.0));
}
```

- Helper not the actual channel

- Physical attributes

  - Propagation loss and delay

  - Fading (Nakagami with equal m values is equivalent to Rayleigh)

Walkthrough of Example Script (cont)

```
WifiHelper wifi = WifiHelper::Default ();
wifi.SetStandard(ns3::WIFI_PHY_STANDARD_80211g);
wifi.SetRemoteStationManager ("ns3::AarfWifiManager");
NetDeviceContainer apDevices;
apDevices = wifi.Install (phy, mac, wifiApNode);
```

- WiFi Helper

  - Creates the Net Devices mentioned earlier

  - Using for setting standards and station manger

  - Sets MAC and PHY information for nodes

Walkthrough of Example Script (cont)

```
MobilityHelper mobility;
mobility.SetPositionAllocator ("ns3::GridPositionAllocator",
                                "MinX", DoubleValue (0.0),
                                "MinY", DoubleValue (0.0),
                                "DeltaX", DoubleValue (5.0),
                                "DeltaY", DoubleValue (10.0),
                                "GridWidth", UintegerValue (3),
                                "LayoutType", StringValue ("RowFirst"));
mobility.SetMobilityModel ("ns3::ConstantPositionMobilityModel");
mobility.Install (wifiApNode);
```

- Used to Position Nodes

- Can be used to specify movement of nodes

Walkthrough of Example Script (cont)

```
OnOffHelper onoff ("ns3::UdpSocketFactory", Address ());
std::string dataRate = "20Mib/s";
onoff.SetConstantRate(dataRate, (uint32_t)1024);
remoteAddress (InetSocketAddress (apAddress.GetAddress (0), 8000));
onoff.SetAttribute ("Remote", remoteAddress);
ApplicationContainer apps = onoff.Install (wifiStaNodes.Get (0));
UniformVariable var;
apps.Start(var.GetValue(0, 0.1));
apps.Stop (Seconds (10.0));
PacketSinkHelper sink("ns3::UdpSocketFactory",
                     InetSocketAddress (apAddress.GetAddress (0), 8000));
apps.Add(sink.Install(wifiApNode.Get(0)));
```

- Applications send (onoff) or receive (sink) packets

Walkthrough of Example Script (cont)

```cpp
flowmon = flowmonHelper.InstallAll();

//Simulator configuration and run commands see next slide
flowmon->CheckForLostPackets ();
Ptr<Ipv4FlowClassifier> classifier
            = DynamicCast<Ipv4FlowClassifier> (flowmonHelper.GetClassifier ());
std::map<FlowId, FlowMonitor::FlowStats> stats
            = flowmon->GetFlowStats ();

for (std::map<FlowId, FlowMonitor::FlowStats>::const_iterator iter = stats.begin (); iter != stats.end ();
++iter){
    Ipv4FlowClassifier::FiveTuple t = classifier->FindFlow (iter->first);
    if (t.sourceAddress == Ipv4Address("10.1.1.1") && t.destinationAddress == Ipv4Address("10.1.1.2")){
        NS_LOG_UNCOND("Throughput in Kib/s over the run time: "
                        << iter->second.rxBytes * 8.0 / (10 * 1024)
                        << std::endl);
    }
}
```

- Monitors all data flows between nodes

- Iterate through statistics to find nodes of interest by IP

Walkthrough of Example Script (cont)

```
Simulator::Stop (Seconds (10.0));
Simulator::Run ();
Simulator::Destroy ();
```

- Schedule the stop time for the simulator

- Run the simulation

- Clean up afterwards

Running the Script

1. Copy Files to :
ns-allinone-3.18/ns-3.18/scratch

2. From the ns-3.18 directory build the simulation with
./waf

3. Run the simulation using waf (remember to remove the .cc from the filename)
   ◦ If no command line arguments are required

       ./waf --run scratch/scriptname

   ◦ If command line arguments are required

       ./waf --run "scratch/scriptname --argument=value"

Questions?