

ESCUELA DE CIENCIAS EXACTAS E INGENIERÍA
Sistemas Complejos
Joaquín Fernando Sanchez Cifuentes
Simulación de red con dos nodos y un enlace wifi en NS-3

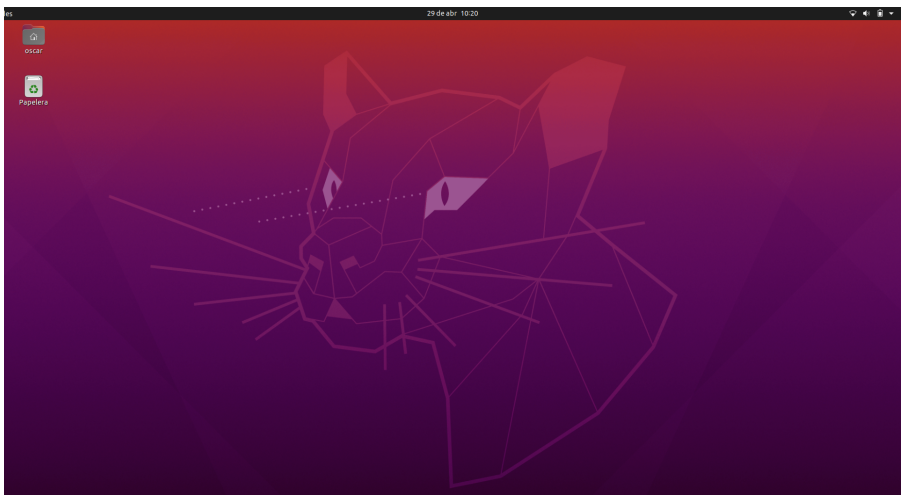
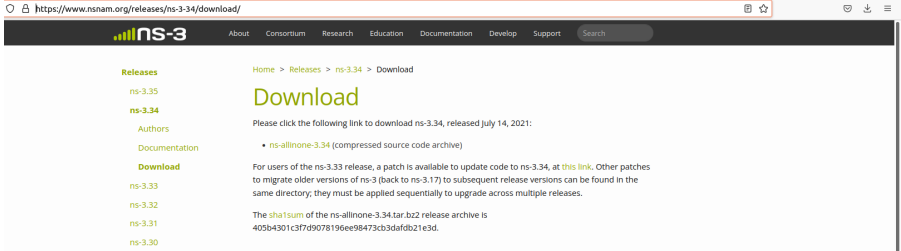


Integrantes:

- Oscar Julian Reyes Torres
- Valeria Bermudez Carvajal
- Santiago Gutierrez Orjuela

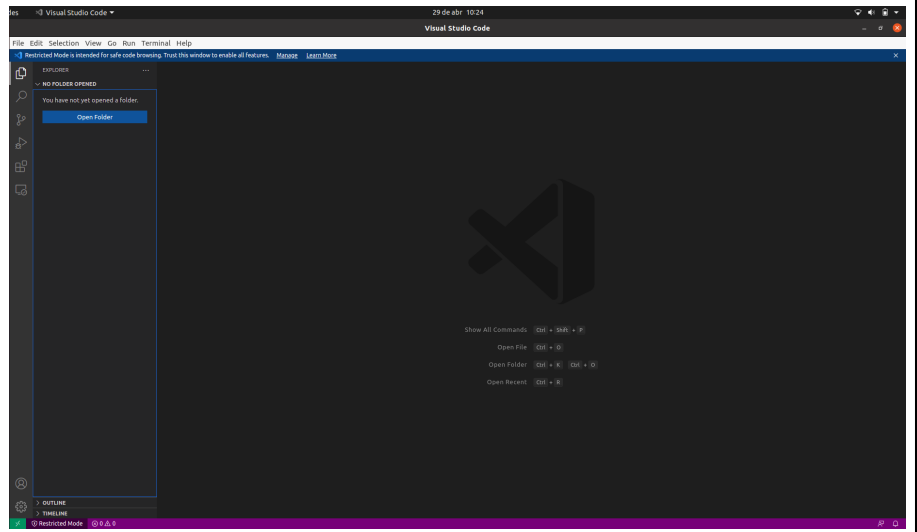
Para esta simulación, se utilizó el simulador ns en su versión 3, donde se realizó un script el cual se imitaba una red con dos nodos y un enlace wifi, programado en lenguaje c++.

Los elementos de software necesarios para realizar esta simulación fueron:

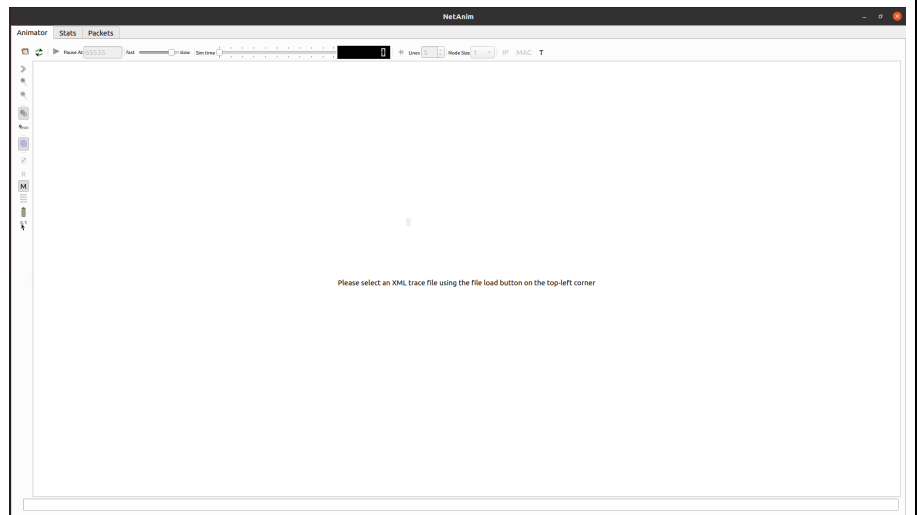
<p>Linux en su distribución Ubuntu 20.04</p>	
<p>Software ns3, en nuestro caso la version 3.34 para linux, a través de la pagina web www.nsnam.org/releases/ns-3-34/download</p>	

ESCUELA DE CIENCIAS EXACTAS E INGENIERÍA
Sistemas Complejos
Joaquín Fernando Sanchez Cifuentes
Simulación de red con dos nodos y un enlace wifi en NS-3

Visual Studio Code, o cualquier editor de texto



Netanim



El script elaborado es el siguiente:

ESCUELA DE CIENCIAS EXACTAS E INGENIERÍA
Sistemas Complejos
Joaquín Fernando Sanchez Cifuentes
Simulación de red con dos nodos y un enlace wifi en NS-3



```
Red2022.cc x
home > oscar > Descargas > ns-allinone-3.34 > ns-3.34 > scratch > Red2022.cc
1  /* -*- Mode:C++; c-file-style:"gnu"; indent-tabs-mode:nil; -*-
2  Autores: Oscar Julian Reyes Torres, Valeria Bermudez Carvajal, Santiago Gutierrez Orjuela
3  Profesor: Joaquín Fernando Sanchez Cifuentes
4  Tema: Simulación de red con dos nodos y un enlace wifi en NS-3
5  Fecha: 29 de abril de 2022
6  */
7
8
9  //Librerías necesarias para su correcto funcionamiento
10 #include "ns3/core-module.h" //Librería de núcleo
11 #include "ns3/network-module.h" //Librería de red
12 #include "ns3/applications-module.h"
13 #include "ns3/mobility-module.h" //Librería de movilidad
14 #include "ns3/internet-module.h"
15 #include "ns3/yans-wifi-helper.h" //Librería Wifi
16 #include "ns3/ssid.h"
17 #include "ns3/netanim-module.h" //Librería de simulación con NetAnim
18
19 // Default Network Topology
20 //
21 // Wifi 10.1.3.0
22 //      *      *      *      *
23 //      |      |      |      |
24 //      n1     n2     n3     n0   10.1.1.0
25 //
26
27 using namespace ns3; //Etiquetas de c++ para ns3
28
29 NS_LOG_COMPONENT_DEFINE ("ThirdScriptExample");
30
31
32 //Función principal
33 int
34 main (int argc, char *argv[])
35 {
36     bool verbose = true;
37     uint32_t nPkt = 1; //Numero de paquetes
38     uint32_t nWifi = 2; //Numero de Nodos wifi móviles
39     bool tracing = false;
40
41     CommandLine cmd (__FILE__);
42     cmd.AddValue ("nPkt", "Numero de Paquetes", nPkt); //Estructura para utilizar directamente en la línea de comandos o terminal para setear valores
43     cmd.AddValue ("nWifi", "Numero de dispositivos wifi", nWifi); //Numero de nodos wifi móviles
44     cmd.AddValue ("verbose", "Decir a las aplicaciones de eco que inicien sesión si es cierto", verbose);
45     cmd.AddValue ("tracing", "Habilitar seguimiento de pcap", tracing); //Si se desea guardar capturas de tráfico que se pueden leer como Wireshark
46
47     cmd.Parse (argc, argv);
48
49     if (nWifi > 250) //Se conficiona a que si se crean mas de 250 nodos móviles retorne un error, dado que el máximo de direcciones ip disponibles son 250
50     {
51         std::cout << "Los nodos móviles nWifi no pueden exceder los 250, dado que se acabarían las direcciones ip para esta simulación" << std::endl;
52         return 1;
53     }
54
55     if (verbose) //Si se activa esta opción veremos:
56     {
57         LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO); //Lo que sucede con la aplicación udp del cliente
58         LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO); //Lo que sucede con la aplicación udp del servidor
59     }
60
61     NodeContainer wifiStaNodes; //Se crean los nodos y se le asigna el nombre de wifiStaNodes -> STA por ser estación (móviles)
62     wifiStaNodes.Create (nWifi); //Creación de estaciones móviles haciendo uso de la variable nWifi por defecto y según requerimiento es de 2
63     NodeContainer wifiApNode;
64     wifiApNode.Create (1); //Se crea el nodo de Access point (solo hay 1)
65
66     YansWifiChannelHelper channel = YansWifiChannelHelper::Default (); //Configurar el canal
67     YansWifiPhyHelper phy; //Configurar la interfaz física del wifi
68     phy.SetChannel (channel.Create ());
69
70     WifiHelper wifi;
71     wifi.SetRemoteStationManager ("ns3::AarWifiManager"); //Configurar la estación
72
73     WifiMacHelper mac;
74     Ssid ssid = Ssid ("SistemasComplejos2022"); //Nombre de la red wifi
75     mac.SetType ("ns3::StaWifiMac", //Se asigna la configuración de la capa Mac
76                 "Ssid", SsidValue (ssid),
77                 "ActiveProbing", BooleanValue (false));
78
79     NetDeviceContainer staDevices; //Dispositivos que se instalan (tarjetas de red y controladores) de cada nodo
80     staDevices = wifi.Install (phy, mac, wifiStaNodes); //Se instala la conexión física, la dirección Mac en los nodos wifi
81
82     mac.SetType ("ns3::ApWifiMac", //Se instala la tarjeta en los nodos de Access point
83                 "Ssid", SsidValue (ssid)); //Se instala la conexión física, la dirección Mac en el Access Point
84
85     NetDeviceContainer apDevices;
86     apDevices = wifi.Install (phy, mac, wifiApNode);
87
88     MobilityHelper mobility; //Se crea la Movilidad
89     mobility.SetPositionAllocator ("ns3::GridPositionAllocator",
```

ESCUELA DE CIENCIAS EXACTAS E INGENIERÍA
Sistemas Complejos
Joaquín Fernando Sanchez Cifuentes
Simulación de red con dos nodos y un enlace wifi en NS-3



```
92      "MinX", DoubleValue (1.1), //Coordenadas minimas en X
93      "MinY", DoubleValue (1.1), //Coordenadas minimas en Y
94      "DeltaX", DoubleValue (5.0), //Lo que se va a mover en X
95      "DeltaY", DoubleValue (10.0), //Lo que se va a mover en Y
96      "GridWidth", UIntegerValue (5), //El ancho
97      "LayoutType", StringValue ("RowFirst")); //Se le asigna la posicion dependiendo de la grilla
98
99      mobility.SetMobilityModel ("ns3::RandomWalk2dMobilityModel", //Sea aleatorio la movilidad
100      "Bounds", RectangleValue (Rectangle (-50, 50, -50, 50)));
101      mobility.Install (wifiStaNodes); //Se instala en los nodos mobiles "wifiStaNodes"
102
103      mobility.SetMobilityModel ("ns3::ConstantPositionMobilityModel"); //Se declara una posicion constante para el Access Point
104      mobility.Install (wifiApNode);
105
106      InternetStackHelper stack; //Pila de protocolos de internet que creamos
107      stack.Install (wifiApNode); //Se instala en el nodo creado como Access Point
108      stack.Install (wifiStaNodes); //Se instala en los nodos creados en este caso los nodos mobiles wifi
109
110      Ipv4AddressHelper address; //Se hace uso del protocolo de enrutamiento ipv4
111
112      //Asigna direcciones ipv4 a los nodos empezando por los nodos mobiles, por lo tanto el nodo 0 va a tener la direccion 10.1.3.1
113      address.SetBase ("10.1.3.0", "255.255.255.0");
114
115      Ipv4InterfaceContainer wifiInterfaces;
116      wifiInterfaces=address.Assign (staDevices);
117      address.Assign (apDevices);
118
119      UdpEchoServerHelper echoServer (9); //Se invoca el servicio
120
121      ApplicationContainer serverApps = echoServer.Install (wifiStaNodes.Get (0)); //Para el servidor el contenedor de la Aplicacion
122      serverApps.Start (Seconds (1.0)); //Para que inicie en 1 segundo
123      serverApps.Stop (Seconds (10.0)); //Para que termine en 10 segundos
124
125      UdpEchoClientHelper echoClient (wifiInterfaces.GetAddress (0), 9); //Se incluye el trafico para el cliente
126      echoClient.SetAttribute ("MaxPackets", UIntegerValue (nPkt)); //Se define el numero de paquetes
127      echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0))); //El intervalo en que se va a transmitir
128      echoClient.SetAttribute ("PacketSize", UIntegerValue (1024)); //El tamaño de paquetes
129
130      //Se crea la aplicacion a que leera el cliente que se consume cada 2 segundos mientras que parsea el servidor es cada 1 segundo
131      ApplicationContainer clientApps =
132      { echoClient.Install (wifiStaNodes.Get (nWifi - 1));
133      clientApps.Start (Seconds (2.0));
134      clientApps.Stop (Seconds (10.0)); //Tambien se termina a los 10 segundos
135
136      Simulator::Stop (Seconds (10.0)); //Detener la simulacion
137
138      //Habilitar la captura del trafico para ser leida por ejemplo con wireshark
139      if (tracing == true)
140      {
141          phy.EnablePcap ("third", apDevices.Get (0));
142      }
143
144      //Se crea el archivo .xml para su visualizacion en NetAnim
145      AnimationInterface anim("Red2022.xml");
146
147      //Constructores del Programa
148      Simulator::Run ();
149      Simulator::Destroy ();
150      return 0;
151
152
```

Una vez realizado el scrip, se crea el archivo .xml, el cual traerá cargada toda la simulación implementada en código. Para esto hacemos uso desde la terminal y utilizamos el comando **“./waf --run scratch/Red2022”**

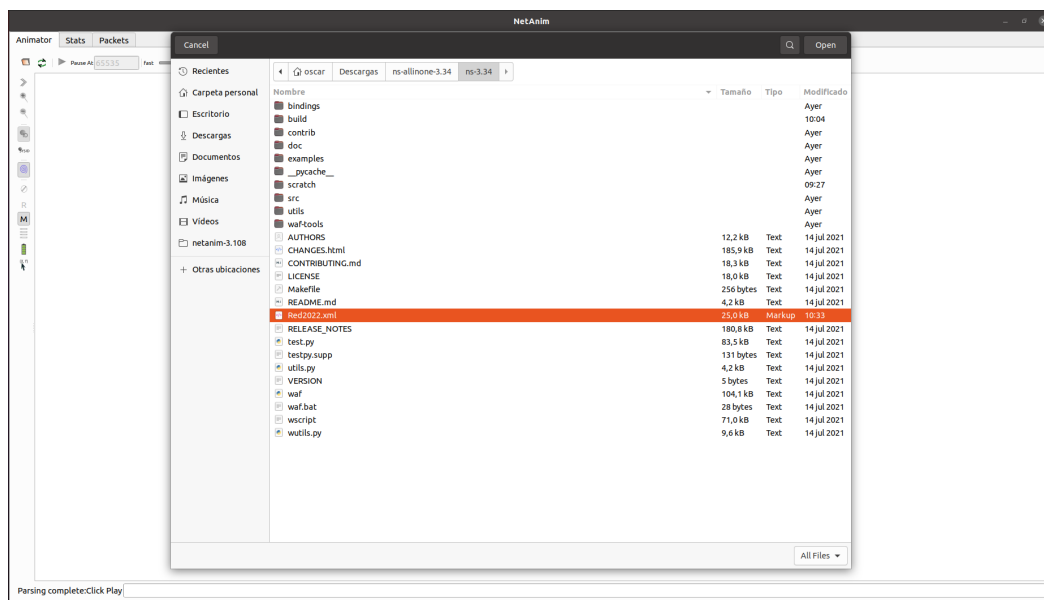
```
oscar@oscar-NBLK-WAX9X: ~/Descargas/ns-allinone-3.34/ns-3.34
oscar@oscar-NBLK-WAX9X:~/Descargas/ns-allinone-3.34/ns-3.34$ ./waf --run scratch/Red2022
Waf: Entering directory `/home/oscar/Dscargas/ns-allinone-3.34/ns-3.34/build'
Waf: Leaving directory `/home/oscar/Dscargas/ns-allinone-3.34/ns-3.34/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (0.542s)
At time +2s client sent 1024 bytes to 10.1.3.1 port 9
At time +2.006s server received 1024 bytes from 10.1.3.2 port 49153
At time +2.006s server sent 1024 bytes to 10.1.3.2 port 49153
At time +2.01599s client received 1024 bytes from 10.1.3.1 port 9
oscar@oscar-NBLK-WAX9X:~/Descargas/ns-allinone-3.34/ns-3.34$
```

ESCUELA DE CIENCIAS EXACTAS E INGENIERÍA
Sistemas Complejos
Joaquín Fernando Sanchez Cifuentes
Simulación de red con dos nodos y un enlace wifi en NS-3

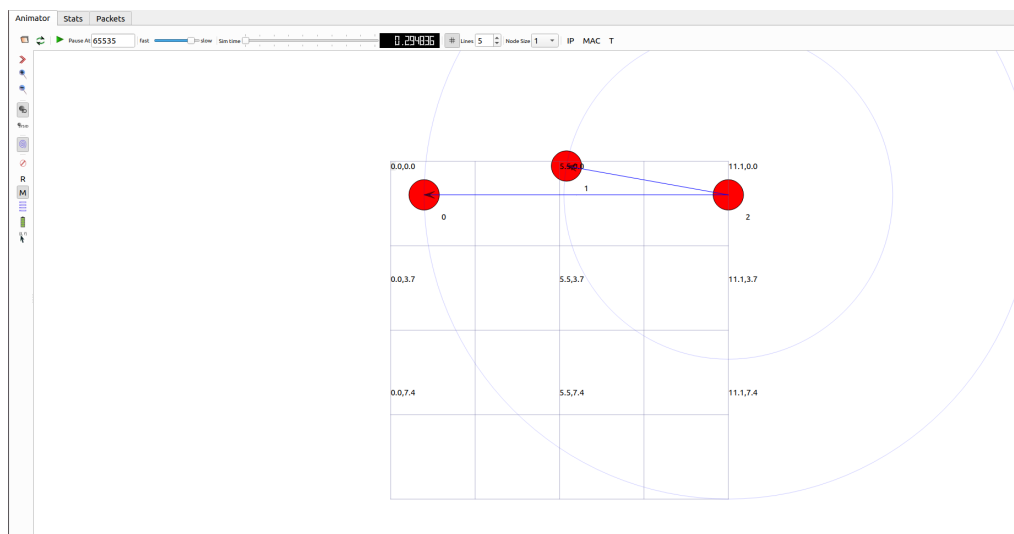


Una vez realizado lo anterior ejecutamos el software netanim y una vez dentro abrimos nuestro .xml llamado “Red2022..xml”

```
oscar@oscar-NBLK-WAX9X: ~/Descargas/ns-allinone-3.34/net...  
oscar@oscar-NBLK-WAX9X:~/Descargas/ns-allinone-3.34/netanim-3.108$ ./NetAnim
```



Como podemos observar a continuación, este simula la red programada con anterioridad y crea la animación:



ESCUELA DE CIENCIAS EXACTAS E INGENIERÍA
Sistemas Complejos
Joaquín Fernando Sanchez Cifuentes
Simulación de red con dos nodos y un enlace wifi en NS-3

En el siguiente enlace se puede observar con mejor detalle la animación: [enlace](#)

Una vez terminada la simulación, dentro del netanim, nos muestra más a detalle el análisis de resultados de la red, extrayendolos información precisa como el detalle de cada nodo, el cual como se puede evidenciar a continuación nos refleja el nombre, la dirección Ipv4 y la dirección MAC utilizadas y programadas en el script:

Node:0 IP: 10.1.3.1 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:01	Node:1 IP: 127.0.0.1 10.1.3.2 IPv6: ::1 MAC: 00:00:00:00:00:02	Node:2 IP: 10.1.3.3 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:03
--	--	--

También nos genera el tráfico de paquetes entre cada nodo



Además, dada la programación utilizada también nos exporta un archivo plano con la cantidad de envíos, nodos destino, nodo final y tiempo utilizado en el envío de cada paquete como se evidencia a continuación:

ESCUELA DE CIENCIAS EXACTAS E INGENIERÍA
Sistemas Complejos
Joaquín Fernando Sanchez Cifuentes
Simulación de red con dos nodos y un enlace wifi en NS-3

Export Table

	From Id	To Id	Tx	Meta
1	2	1	0.090036	
2	2	0	0.090036	
3	1	2	0.120034	
4	2	1	0.120162	
5	2	0	0.120162	
6	2	1	0.12024	
7	2	0	0.12024	
8	1	0	0.12034	
9	1	2	0.12034	
10	0	1	0.120652	
11	0	2	0.120652	
12	2	1	0.12078	
13	2	0	0.12078	
14	2	1	0.120858	
15	2	0	0.120858	
16	0	1	0.120958	
17	0	2	0.120958	
18	2	1	0.192436	
19	2	0	0.192436	
20	2	1	0.294836	
21	2	0	0.294836	
22	2	1	0.397236	
23	2	0	0.397236	
24	2	1	0.499636	
25	2	0	0.499636	
26	2	1	0.602036	
27	2	0	0.602036	
28	2	1	0.704436	
29	2	0	0.704436	
30	2	1	0.806836	
31	2	0	0.806836	
32	2	1	0.909236	
33	2	0	0.909236	
34	2	1	1.01164	
35	2	0	1.01164	
36	2	1	1.11404	
37	2	0	1.11404	
38	2	1	1.21644	
39	2	0	1.21644	

Vale la pena mencionar y haciendo una verificación a la tabla generada, podemos observar que en ocasiones nuestro Access Point no fue quien envió los paquetes dado que en algunos casos, como en los registros 3, 8,9,10, 11, entre otros, fueron diferentes nodos quienes entablaron una comunicación y un envío de información siendo este un tráfico variable.

A través del siguiente enlace se encuentra el repositorio donde está el script programado para su verificación y ejecución.

[click aquí](#)