

**ESCUELA DE CIENCIAS EXACTAS E INGENIERÍA**  
**Sistemas Complejos**  
**Joaquín Fernando Sanchez Cifuentes**  
**Simulación de red Ad Hoc**



**Integrantes:**

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

Para esta primera simulación de nuestra red ad-hoc se creó un script que se evidencia a continuación y que está disponible en el siguiente [enlace](#), el cual hace uso de las librerías de ns3:

```
RedAdHoc.cpp X
home > oscar > Descargas > ns-allinone-3.34 > ns-3.34 > scratch > RedAdHoc.cpp
1  /*
2  3  Autores: Oscar Julian Reyes Torres, Valeria Bermudez Carvajal, Santiago Gutierrez Orjuela
4  Profesor: Joaquín Fernando Sanchez Cifuentes
5  Tema: Simulación de red Ad-Hoc
6  Fecha: 01 de mayo de 2022
7  */
8
9  /*Importa librerías*/
10 #include "ns3/mobility-module.h" //Librería de movilidad
11 #include "ns3/wifi-module.h" //Librería Wifi
12 #include "ns3/core-module.h" //Librería de núcleo
13 #include "ns3/network-module.h"
14 #include "ns3/internet-module.h"
15 #include "ns3/applications-module.h"
16
17 using namespace ns3; //Etiquetas de c++ para ns3
18
19 NS_LOG_COMPONENT_DEFINE ("red Adhoc");
20
21 int main (int argc, char *argv[])
22 {
23     int verbose=1;
24
25     //Activar trazas de dispositivos
26     WifiHelper wifi;
27
28     if(verbose){
29         wifi.EnableLogComponents();
30     }
31     wifi.SetStandard(WIFI_PHY_STANDARD_80211b);
32
33     //Creación de 2 nodos
34     NodeContainer nodes;
35     nodes.Create(2);
36
37     //Elección capa física y MAC
38     YansWifiPhyHelper wifiPhy = YansWifiPhyHelper::Default();
39     NqosWifiMacHelper wifiMac = NqosWifiMacHelper::Default();
40     wifiPhy.SetRemoteStationManager("ns3::ConstantRateWifiManager", "DataMode",StringValue (phyMode), "ControlMode",StringValue (phyMode));
41     wifiMac.SetType("ns3::AdhocWifiMac");
42
43     YansWifiPhyHelper wifiChannel;
44     wifiChannel.SetPropagationDelay("ns3::ConstantSpeedPropagationDelayModel");
45     wifiChannel.AddPropagationLoss("ns3::FixedRssLossModel", "Rss", DoubleValue(rss));
46     wifiPhy.SetChannel (wifiChannel.Create());
47     NetDeviceContainer devices = wifi.Install(wifiPhy, wifiMac,nodes);
48
49     //Ajustes de Movilidad
50     MobilityHelper mobility;
51
52     //Posición Inicial
53     Ptr<ListPositionAllocator> positionAlloc = CreateObject<ListPositionAllocator> ();
54     positionAlloc->Add (Vector(0.0,0.0,0.0));
55     positionAlloc->Add (Vector(5.0,0.0,0.0));
56     mobility.SetPositionAllocator (positionAlloc);
57
58     //Elección del modelo de movilidad
59     mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
60     mobility.Install (c);
61
62     //Trazas
63     AsciiTraceHelper ascii;
64     wifiPhy.EnableAsciiAll(ascii.CreateFileStream("net-adhoc.tr"));
65     wifiPhy.EnablePcap("net-adhoc", devices);
66
67     //Constructores del Programa
68     Simulator::Run();
69     Simulator::Destroy();
70
71 }
```

**ESCUELA DE CIENCIAS EXACTAS E INGENIERÍA**  
**Sistemas Complejos**  
**Joaquín Fernando Sanchez Cifuentes**  
**Simulación de red Ad Hoc**



Lamentablemente, para esta simulación se vio obstruida por el módulo *NqosWifiMacHelper* el cual fue descontinuado y tampoco permitía el establecimiento de la capa MAC para los nodos creados.

La siguiente parte fue la instalación del protocolo de enrutamiento B.A.T.M.A.N. en todos los dispositivos que van a hacer parte de la red y de esta forma empezar con una simulación de ping.

Para la instalación del protocolo B.A.T.M.A.N. se hizo uso del sudo [apt-get install batctl](#) comando desde la terminal de linux en ambos nodos de cómputo, para este caso la distribución de ubuntu 20.04:

```
oscar@oscar-NBLK-WAX9X: ~  
oscar@oscar-NBLK-WAX9X:~$ sudo apt-get install batctl  
[sudo] contraseña para oscar:  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.  
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi libfwupdplugin1 libgstreamer-plugins-bad1.0-0 libva-wayland2  
Utilice «sudo apt autoremove» para eliminarlos.  
Se instalarán los siguientes paquetes NUEVOS:  
  batctl  
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 2 no actualizados.  
Se necesita descargar 74,3 kB de archivos.  
Se utilizarán 199 kB de espacio de disco adicional después de esta operación.  
Des:1 http://co.archive.ubuntu.com/ubuntu focal/universe amd64 batctl amd64 2019.5-1 [74,3 kB]  
Descargados 74,3 kB en 1s (81,6 kB/s)  
Seleccionando el paquete batctl previamente no seleccionado.  
(Leyendo la base de datos ... 197360 ficheros o directorios instalados actualmente.)  
Preparando para desempaquetar .../batctl_2019.5-1_amd64.deb ...  
Desempaquetando batctl (2019.5-1) ...  
Configurando batctl (2019.5-1) ...  
Procesando disparadores para man-db (2.9.1-1) ...  
oscar@oscar-NBLK-WAX9X:~$
```

Posteriormente, se creó un script que configura el enrutamiento en este protocolo, el cual reposa en el siguiente [enlace](#)

Vale la pena mencionar que para compilar y ejecutar los script anteriormente mencionados, es importante definir la dirección ip correspondiente a cada nodo de cómputo, para esto, se consulta el nombre y las propiedades de las interfaces de cada computador.

Para el primer nodo de cómputo desde la terminal usamos el comando [ifconfig](#) reflejando estar conectado a una red Wifi doméstica:

**ESCUELA DE CIENCIAS EXACTAS E INGENIERÍA**  
**Sistemas Complejos**  
**Joaquín Fernando Sanchez Cifuentes**  
**Simulación de red Ad Hoc**



```
oscar@oscar-NBLK-WAX9X: ~  
oscar@oscar-NBLK-WAX9X:~$ ifconfig  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0x10<host>  
    loop txqueuelen 1000 (Bucle local)  
    RX packets 23907 bytes 2396755 (2.3 MB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 23907 bytes 2396755 (2.3 MB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
wlp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.168.0.23 netmask 255.255.255.0 broadcast 192.168.0.255  
    inet6 fe80::708a:7174:fed8:778 prefixlen 64 scopeid 0x20<link>  
    inet6 2800:484:4c81:6329:372:2118:1b87:c94b prefixlen 64 scopeid 0x0<global>  
    inet6 2800:484:4c81:6329:fa05:ce58:adaa:ce6f prefixlen 64 scopeid 0x0<global>  
    ether 64:6c:80:a1:3d:0b txqueuelen 1000 (Ethernet)  
    RX packets 103452 bytes 77195732 (77.1 MB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 64547 bytes 28734404 (28.7 MB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
oscar@oscar-NBLK-WAX9X:~$
```

Posteriormente, haciendo uso del comando [iwconfig](#) se observa la interfaz inalámbrica con el protocolo IEEE 802.11 mediante un access point.

```
oscar@oscar-NBLK-WAX9X: ~  
oscar@oscar-NBLK-WAX9X:~$ iwconfig  
lo          no wireless extensions.  
  
wlp2s0      IEEE 802.11  ESSID:"JUDIAN 5g"  
    Mode:Managed  Frequency:5.22 GHz  Access Point: F4:95:1B:D5:77:D4  
    Bit Rate=58.5 Mb/s   Tx-Power=23 dBm  
    Retry short limit:7   RTS thr:off   Fragment thr:off  
    Power Management:on  
    Link Quality=36/70  Signal level=-74 dBm  
    Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0  
    Tx excessive retries:0  Invalid misc:324  Missed beacon:0  
  
oscar@oscar-NBLK-WAX9X:~$
```

Dada la información anteriormente recolectada, se procedió a realizar la configuración en ambos dispositivos mediante el script del siguiente [enlace](#)

Un aspecto a tener en cuenta, es que la WLAN0 se cambió por el nombre de acuerdo a el de cada computador, además la dirección IP, se modificó en su último octeto con el fin de diferenciar cada nodo.

**ESCUELA DE CIENCIAS EXACTAS E INGENIERÍA**  
**Sistemas Complejos**  
**Joaquín Fernando Sanchez Cifuentes**  
**Simulación de red Ad Hoc**



- Para el Equipo 1 se asigno la direccion → 192.168.0.3

```
$ configuracionBATMAN.sh
home > oscar > Descargas > AdHoc_Network-master > PC-Huawei > $ configuracionBATMAN.sh
1
2 #Autores: Oscar Julian Reyes Torres, Valeria Bermudez Carvajal, Santiago Gutierrez Orjuela
3 #Profesor: Joaquin Fernando Sanchez Cifuentes
4 #Tema: Simulacion de red Ad-Hoc
5 #Fecha: 01 de mayo de 2022
6 #Configuracion Equipo 1 - Red Ad Hoc (192.168.0.3)
7
8
9 #!/bin/bash
10
11 #Modo
12 modprobe batman-adv
13 #Se para el proseso de red de la maquina
14 sudo pkill NetworkManager
15 #se apaga la interfaz wan de la maquina y se asigna el modo adhoc
16 ifconfig wlan0 down
17 iwconfig wlan0 mode ad-hoc
18
19 ifconfig wlan0 mtu 1532
20 #Asignacion del acces point y el channel
21 iwconfig wlan0 mode ad-hoc essid RED-Adhoc ap 02:1B:55:AD:0C:02 channel 1
22 sleep 1
23 #Se enciende la interfaz wan de la maquina
24 ip link set wlan0 up
25 sleep 1
26 #Se asigna la interfaz virtual de BATMAN a la WAN
27 batctl if add wlan0
28 ifconfig bat0 up
29 #Se asigna la IP a la interfaz virtual
30 ifconfig bat0 192.168.0.3
31
32 echo "Configuracion realizada exitosamente"
```

- Para el Equipo 2 se asigno la direccion → 192.168.0.4

```
Abrir  ▾  ~/Descargas/AdHoc_Network-master/PC-Dell-Valeria  Guardar  -  □  ×
*confBATMAN.sh
1 #Autores: Oscar Julian Reyes Torres, Valeria Bermudez Carvajal, Santiago Gutierrez Orjuela
2 #Profesor: Joaquin Fernando Sanchez Cifuentes
3 #Tema: Simulacion de red Ad-Hoc
4 #Fecha: 01 de mayo de 2022
5 #Configuracion Equipo 1 - Red Ad Hoc (192.168.0.4)
6
7 #!/bin/bash
8
9 #Modo
10 modprobe batman-adv
11 #Se para el proseso de red de la maquina
12 sudo pkill NetworkManager
13 #se apaga la interfaz wan de la maquina y se asigna el modo adhoc
14 ifconfig wlp3s0 down
15 iwconfig wlp3s0 mode ad-hoc
16
17 ifconfig wlp3s0 mtu 1532
18 #Asignacion del acces point y el channel
19 iwconfig wlp3s0 mode ad-hoc essid RED-Adhoc ap 02:1B:55:AD:0C:02 channel 1
20 sleep 1
21 #Se enciende la interfaz wan de la maquina
22 ip link set wlp3s0 up
23 sleep 1
24 #Se asigna la interfaz virtual de BATMAN a la WAN
25 batctl if add wlp3s0
26 ifconfig bat0 up
27 #Se asigna la IP a la interfaz virtual
28 ifconfig bat0 192.168.0.4
29
30 echo "Configuracion realizada exitosamente"
```

A continuación, se procede a ejecutar el script en ambos nodos de cómputo, lo anterior en modo root al cual se accede con el comando `chmod +x configuracionBATMAN.sh`:

```
oscar@oscar-NBLK-WAX9X:~/Descargas/AdHoc_Network-master/PC-Huawei-Oscar$ chmod +x configuracionBATMAN.sh
oscar@oscar-NBLK-WAX9X:~/Descargas/AdHoc_Network-master/PC-Huawei-Oscar$
```

Dado lo anterior se procede a la verificación que la red se configuró correctamente mediante los comandos de [ifconfig](#) e [iwconfig](#)

Se evaluó que la interfaz virtual bat0 se haya creado correctamente con la ip que se indico en cada uno de los dispositivos, además de que la red LAN está caída.

```
valc@valc-Aspire-A315-51:~/Documentos/AdHoc_Network$ ifconfig
bat0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.4 netmask 255.255.255.0 broadcast 192.168.0.255
    inet6 fe80::9cd0:1ff:fea2:61a2 prefixlen 64 scopeid 0x20<link>
    ether 9e:d0:01:a2:61:a2 txqueuelen 1000 (Ethernet)
    RX packets 12 bytes 504 (504.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 15 bytes 1900 (1.9 KB)
    TX errors 0 dropped 42 overruns 0 carrier 0 collisions 0

enp2s0f1: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether d8:c4:97:8f:0f:d8 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Bucle local)
    RX packets 1380 bytes 111800 (111.8 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1380 bytes 111800 (111.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1532
    inet6 fe80::525b:c2ff:feaa:f591 prefixlen 64 scopeid 0x20<link>
    ether 50:5b:c2:aa:f5:91 txqueuelen 1000 (Ethernet)
    RX packets 1055 bytes 691790 (691.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1073 bytes 210679 (210.6 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Luego podemos evidenciar que la interfaz inalámbrica de cada uno de los dispositivos cambió sus propiedades tal como el modo que debe estar en adhoc, el essid y el access point o cell:

```
valc@valc-Aspire-A315-51:~/Documentos/AdHoc_Network$ iwconfig
lo          no wireless extensions.

enp2s0f1    no wireless extensions.

wlp3s0      IEEE 802.11  ESSID:"RED-Adhoc"
            Mode:Ad-Hoc  Frequency:2.412 GHz  Cell: 02:1B:55:AD:0C:02
            Tx-Power=20 dBm
            Retry short limit:7   RTS thr:off   Fragment thr:off
            Power Management:on

bat0        no wireless extensions.
```



Por otra parte, comprobamos la tabla de enrutamiento del protocolo BATMAN, la cual nos muestra los dispositivos conectados a la red, el resultado fue el siguiente:

```
root@valc-Aspire-A315-51:/home/valc/Documentos/AdHoc_Network# sudo batctl o
[B.A.T.M.A.N. adv 2021.0, MainIF/MAC: wlp3s0/50:5b:c2:aa:f5:91 (bat0/9e:d0:01:a2:61:a2 BATMAN_IV)]
  Originator      last-seen (#/255) NextHop      [outgoingIF]
* 30:52:cb:9a:d0:8f    0.360s    (223) 30:52:cb:9a:d0:8f [ wlp3s0]
```

Por último, realizamos la prueba de ping entre los dispositivos y ambas direcciones, esto con el fin de que los paquetes que se transmiten por la red , están llegando a su destino, además podemos evidenciar propiedades como el origen, destino, saltos y el tiempo de retardo 192.168.0.4 a 192.168.0.3

```
valc@valc-Aspire-A315-51:~/Documentos/AdHoc_Network$ ping 192.168.0.3
PING 192.168.0.3 (192.168.0.3) 56(84) bytes of data.
64 bytes from 192.168.0.3: icmp_seq=1 ttl=64 time=2.22 ms
64 bytes from 192.168.0.3: icmp_seq=2 ttl=64 time=9.39 ms
64 bytes from 192.168.0.3: icmp_seq=3 ttl=64 time=3.56 ms
64 bytes from 192.168.0.3: icmp_seq=4 ttl=64 time=2.70 ms
64 bytes from 192.168.0.3: icmp_seq=5 ttl=64 time=1.41 ms
64 bytes from 192.168.0.3: icmp_seq=6 ttl=64 time=1.54 ms
64 bytes from 192.168.0.3: icmp_seq=7 ttl=64 time=1.80 ms
64 bytes from 192.168.0.3: icmp_seq=8 ttl=64 time=1.18 ms
64 bytes from 192.168.0.3: icmp_seq=9 ttl=64 time=3.04 ms
64 bytes from 192.168.0.3: icmp_seq=10 ttl=64 time=9.24 ms
64 bytes from 192.168.0.3: icmp_seq=11 ttl=64 time=1.41 ms
64 bytes from 192.168.0.3: icmp_seq=12 ttl=64 time=1.23 ms
64 bytes from 192.168.0.3: icmp_seq=13 ttl=64 time=5.43 ms
```

### Referencias

- Armando Mercado, R. B. (12 de junio de 2018). *ACADEMIA*. Obtenido de Redes inalámbricas ad hoc: [Academia.edu](https://academia.edu)
- Hernández, S. M. (20 de septiembre de 2020). *Enrutamiento BABEL y BATMAN en una red Ad Hoc*. Obtenido de Universidad Distrital: [udistrital.edu.co](https://udistrital.edu.co)
- Rosas, M. A. (2020 de noviembre de 03). *Diseño de la Topología de una red Ad Hoc*. Obtenido de Universidad Politecnica de Catalunya: [upcommons.upc.edu](https://upcommons.upc.edu)