Introduction to WSNs

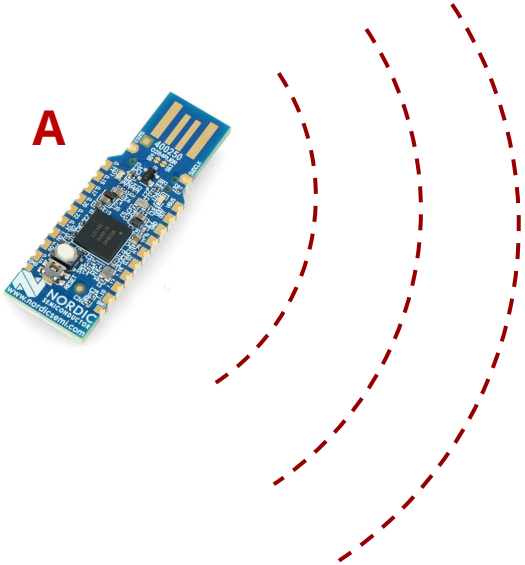
School Year 2023-2024

Valsalice

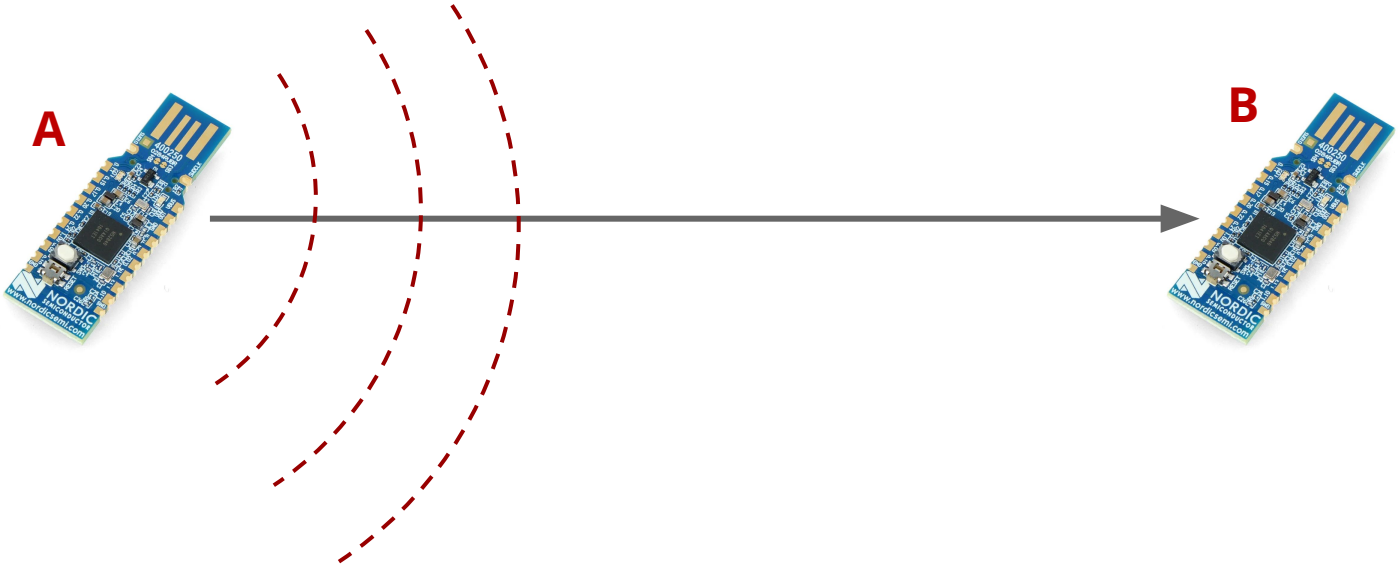
Introductions

- Alberto Spina
  - (2015) Valsalice Alumni
  - (2019) MEng Computing - Imperial College London
  - (2023/current) Software Engineer - London

Live Demo



Live Demo

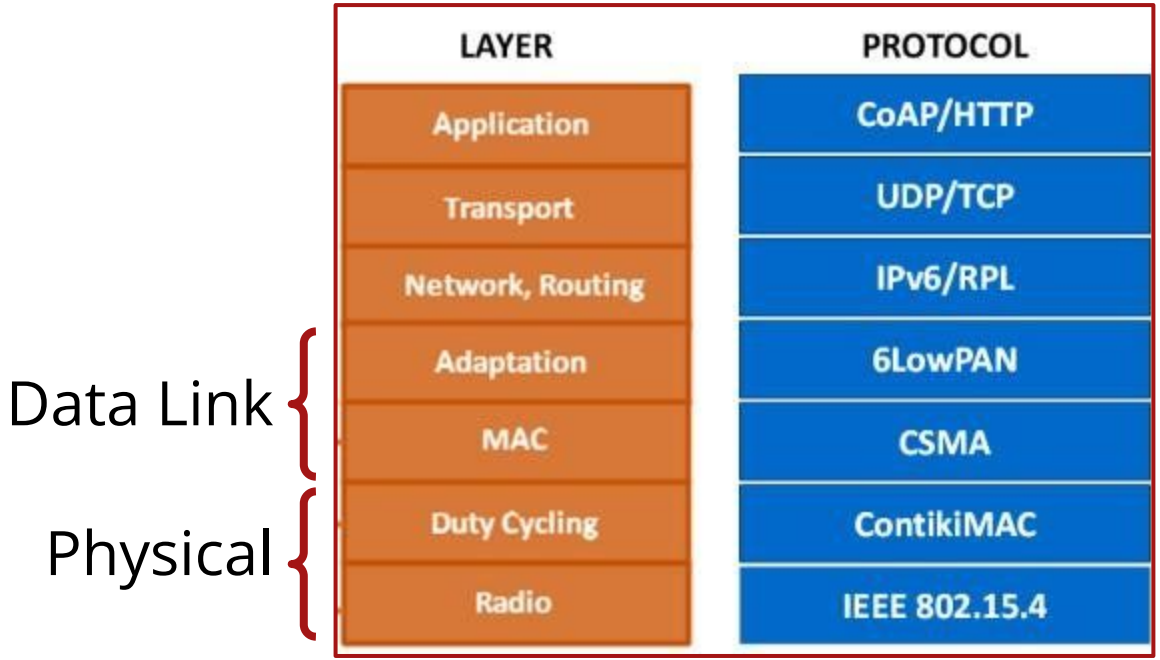


What are Wireless Sensor Networks?

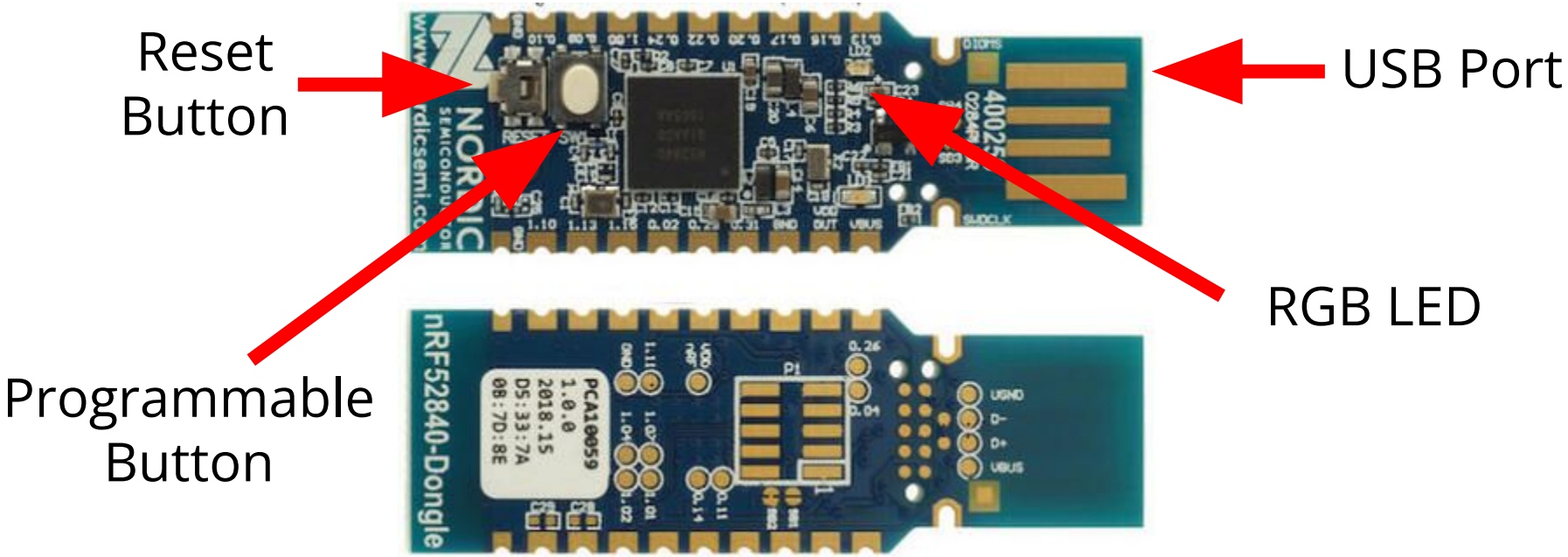
It refers to a **network of interconnected physical devices**, embedded with **sensors**, software, and connectivity, allowing them to **collect and exchange data** over the radio.

What is Contiki?

Contiki is an OS with configurable network layers:



What is the nRF52840?



Lab Objectives

- Program **nRF52840** dongles to disseminate and aggregate data in the real world.
- Use **packet sniffers** to visualise radio communication.
- Explore the use of **Contiki-NG** to program WSNs.
- Learn to use **Cooja** to simulate WSNs.

Open your Virtual Machines

1. Turn on your Laptops
2. Login to Windows using “User”
3. Open the **Virtual Box** program
4. Add a new Virtual Machine (**Ctrl + A**)
5. Open the **VirtualBox** folder ⚠ (**NOT** the .VirtualBox)
6. Select the **nRF52840LAB** file
7. Click **Start**

Prepare the Coding Environment

1. Start the Virtual Machine **nRF52840LAB**
2. Log-in using credentials:

   Username: **ubuntu**

   Password: **ubuntu**

3. Open **Visual Studio Code** (use the App bar on the left)

Prepare the Coding Environment

4 From the Terminal:

```
make setup
```

5

```
○ → valsalice-iot-23 git:(master) make setup
  Enter your username:
```

6

```
Password
```

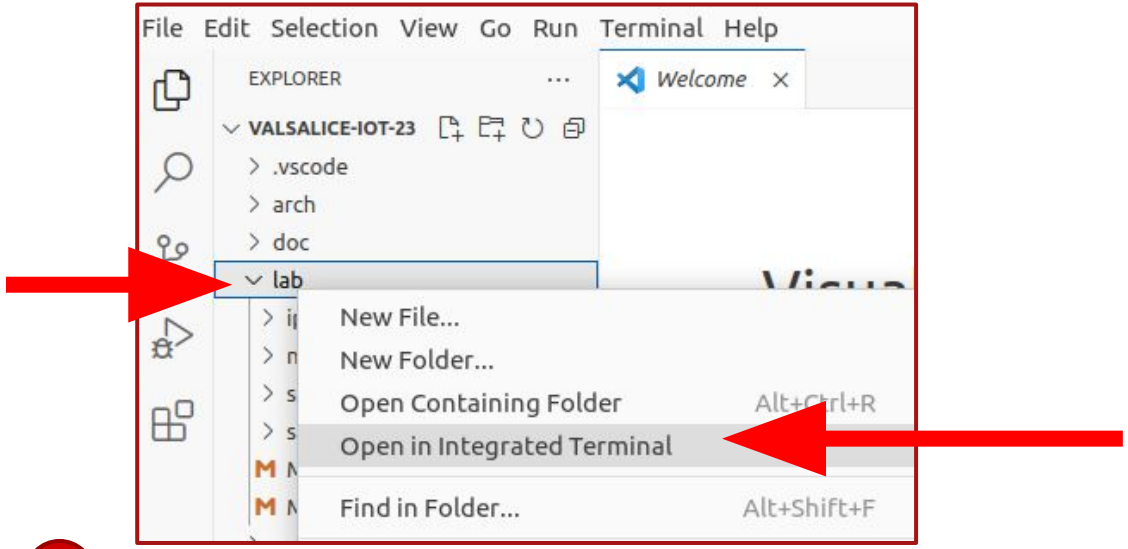
7

```
✅ Repository setup complete!
```

If you see **any (yellow) errors** input the credentials again

Prepare the Coding Environment

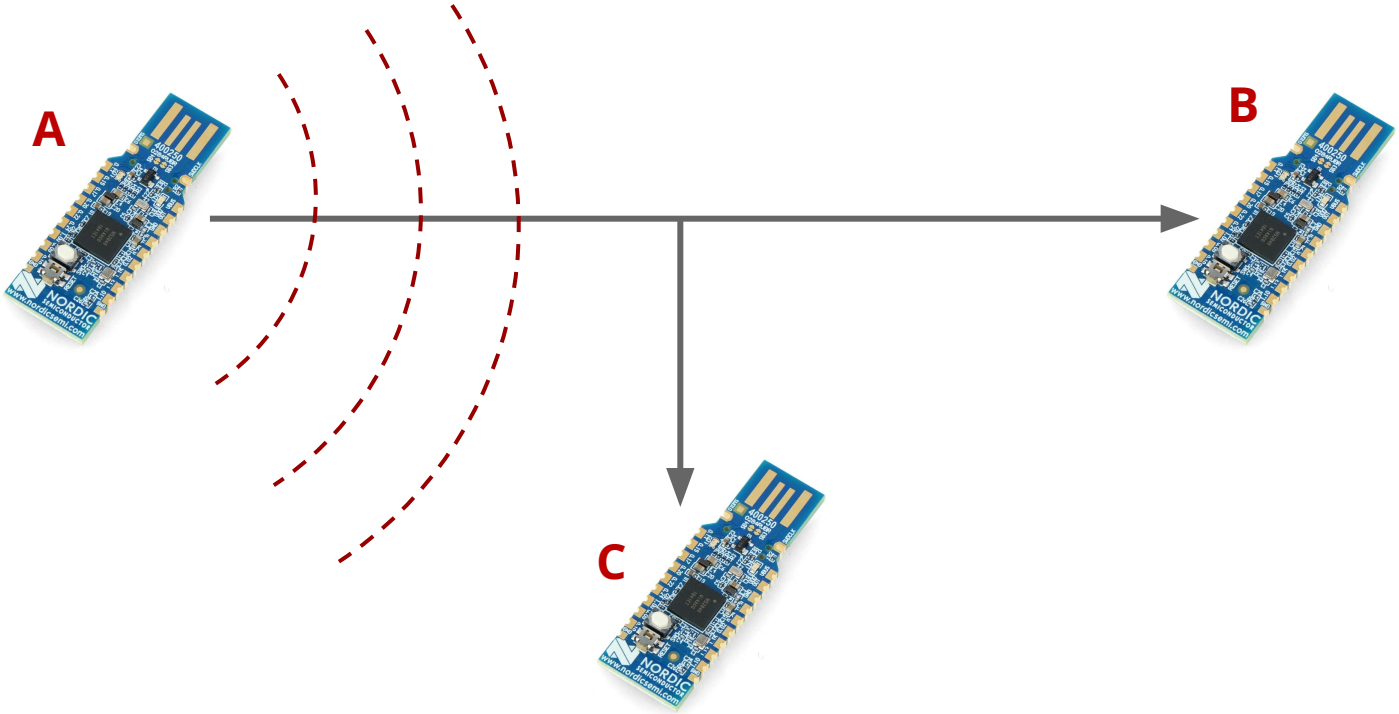
8 Open the **lab** folder in the terminal (**right-click**)



9 You should see the following in the terminal:

```
lab git:(master)
```

Receiving Packet Broadcasts



Receiving Packet Broadcasts

1. From the Terminal go to the **nullnet** directory

```
cd nullnet
```

2. Attach the **nRF52840** chip to your laptops

⚠ Ensure the device is in **bootloader mode** (blinking red light)

3. Program the firmware

```
make receiver.dfu-upload
```

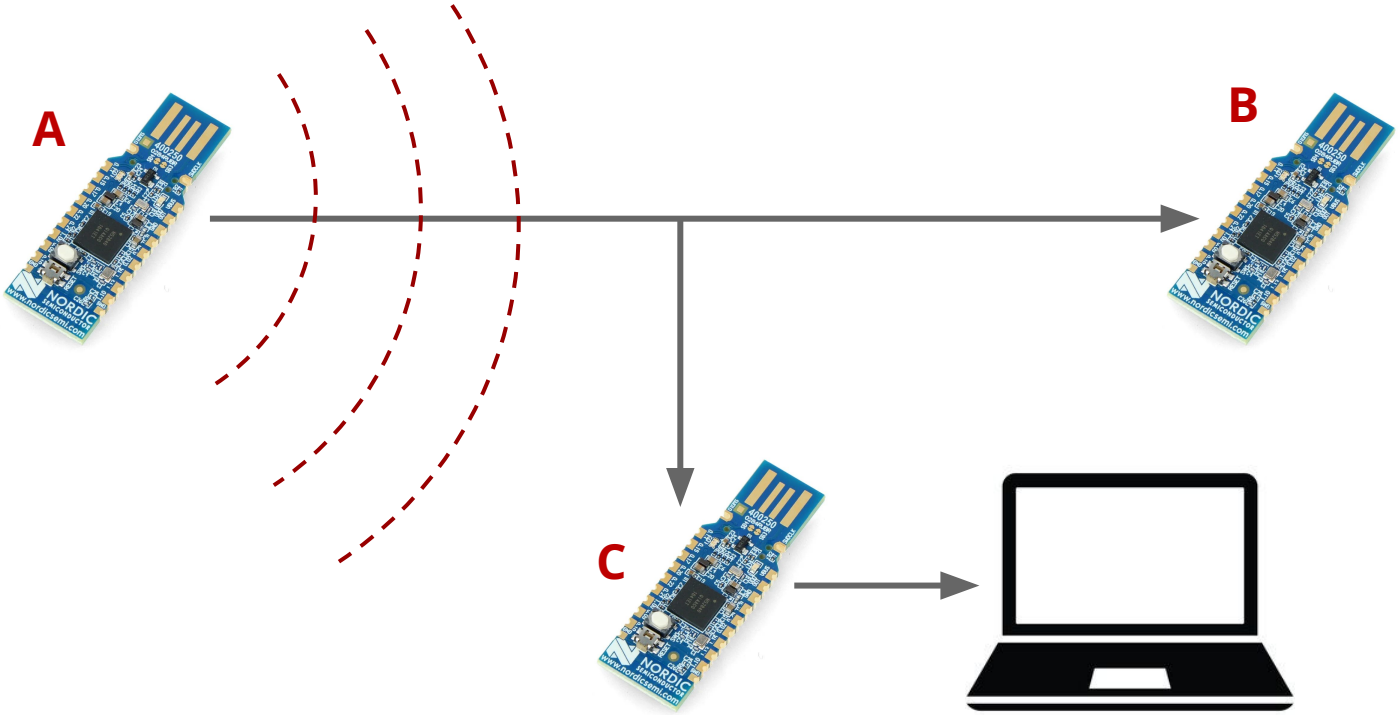
4. Attach to the serial output

```
make login
```

Receiving Packet Broadcasts

```
[INFO: App        ] Polling for messages...
[INFO: App        ] Polling for messages...
[INFO: App        ] Polling for messages...
[INFO: App        ] Polling for messages...
[INFO: App        ] Polling for messages...
[INFO: App        ] Message received:  user: DonBosco; pwd: Valsa
```

Sniffing Packets



Sniffing Packets

1. From the Terminal go to the **nullnet** directory

```
cd ../sensniff
```

2. ⚠ Put the **nRF52840** in **bootloader mode** (blinking red light)

3. Program the firmware

```
make sensniff.dfu-upload
```

4. Start the sniffer

```
make PORT=/dev/ttyACM0 sniff
```

Sniffing Packets

```
using saved target 'nrf52840'
../../tools/sensniff/sensniff.py -b 460800 -d /dev/ttyACM0
Commands:
c: Print current RF Channel
m: Print Min RF Channel
M: Print Max RF Channel
n: Trigger new pcap header before the next frame
h,?: Print this message
<number>: Change RF channel.
q: Quit
Sniffing in channel: 26
Remote end not reading
Remote end not reading
```

Sniffing Packets

1. Open **Wireshark**:
   - a. Click the **Windows Key**
   - b. Search for Wireshark
   - c. Click the Wireshark Icon once

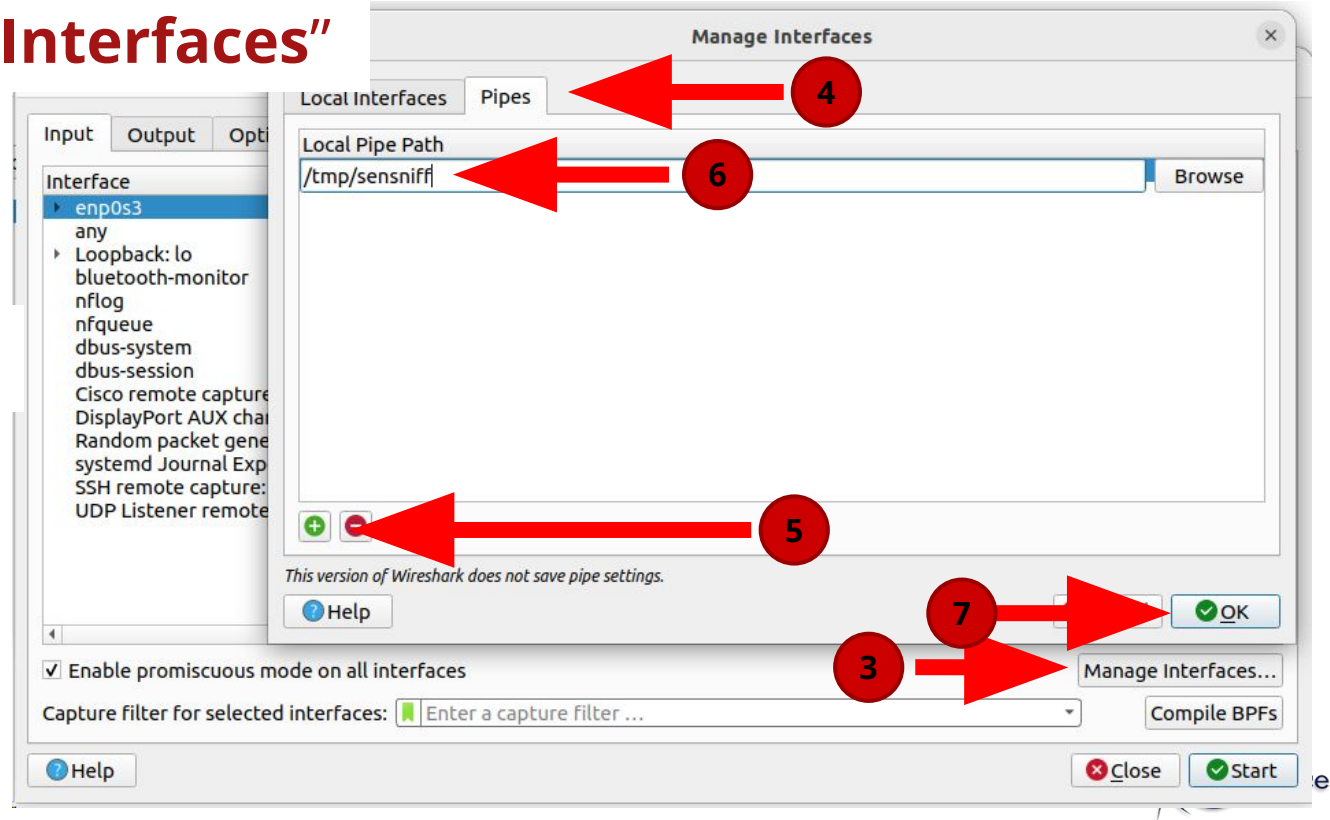
2. Open "**Capture > Options**" by pressing **Ctrl + K**

Sniffing Packets

3. Click "**Manage Interfaces**"
4. Click "**Pipes**"
5. Click "**+**"
6. Write:

   /tmp/sensniff
7. Click "OK"

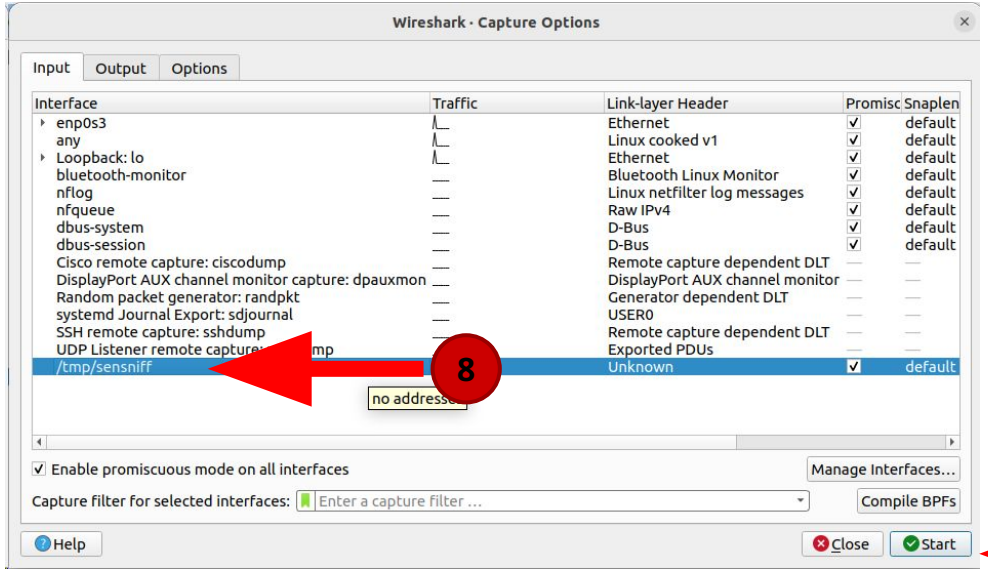
⚠ **only once**



Sniffing Packets

8 Only select **"/tmp/sensniff"**

9 Click "**Start**"



Sniffing Packets

Packets

IP Address Information

Data

Capturing from /tmp/sensniff

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

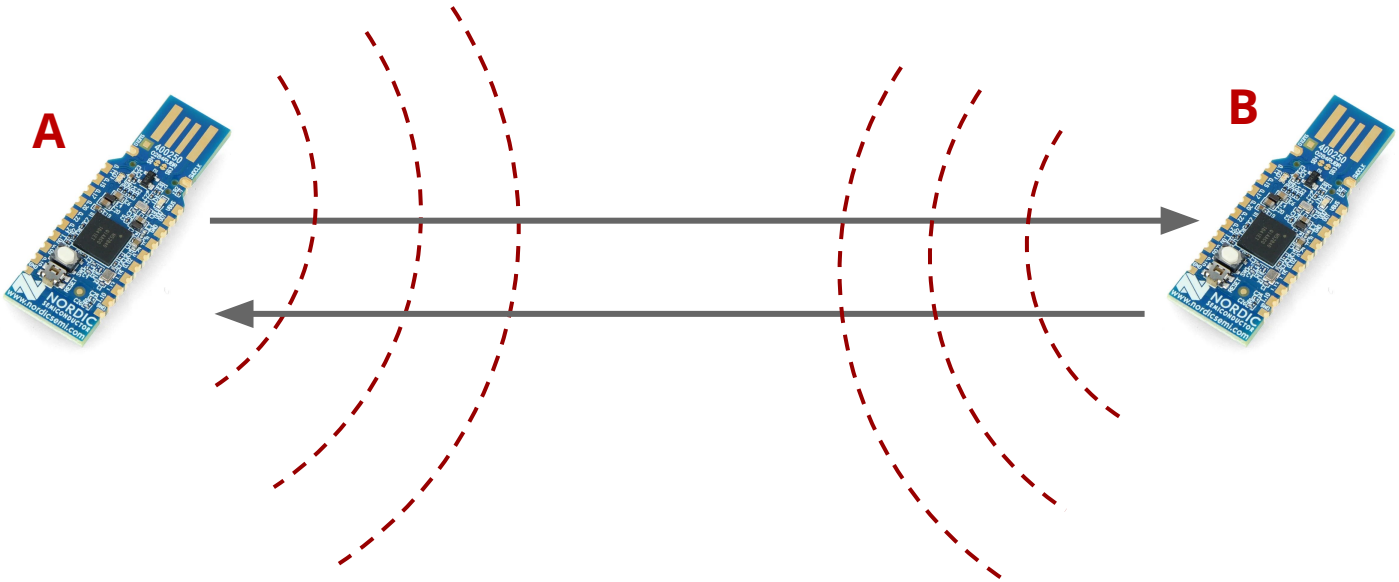
Apply a display filter … <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	f4:ce:36:3e:30:f0:a…	Broadcast	IEEE 8…	47	Data, Dst: Broadcast, Src
2	4.983968	f4:ce:36:3e:30:f0:a…	Broadcast	IEEE 8…	47	Data, Dst: Broadcast, Src
3	9.968567	f4:ce:36:3e:30:f0:a…	Broadcast	IEEE 8…	47	Data, Dst: Broadcast, Src
4	14.983546	f4:ce:36:3e:30:f0:a…	Broadcast	IEEE 8…	47	Data, Dst: Broadcast, Src
5	19.967769	f4:ce:36:3e:30:f0:a…	Broadcast	IEEE 8…	47	Data, Dst: Broadcast, Src
6	24.975419	f4:ce:36:3e:30:f0:a…	Broadcast	IEEE 8…	47	Data, Dst: Broadcast, Src
7	29.974961	f4:ce:36:3e:30:f0:a…	Broadcast	IEEE 8…	47	Data, Dst: Broadcast, Src

```
▸ Frame 2: 47 bytes on wire (376 bits), 47 bytes captured (376 bits) on interface /tmp/sensniff, id 0
▾ IEEE 802.15.4 Data, Dst: Broadcast, Src: NordicSe_3e:30:f0:a3:8e, Bad FCS
  ▸ Frame Control Field: 0xd841, Frame Type: Data, PAN ID Compression, Destination Addressing Mode: Shor…
    Sequence Number: 156
    Destination PAN: 0xabcd
    Destination: 0xffff
    Extended Source: NordicSe_3e:30:f0:a3:8e (f4:ce:36:3e:30:f0:a3:8e)
    FCS: 0xe0bb (Incorrect, expected FCS=0xcee3)
  ▸ [Expert Info (Warning/Checksum): Bad FCS]
▾ Data (30 bytes)
    Data: 20757365723a20446f6e426f73636f3b207077643a2056616c7361202c01
    [Length: 30]

0000  41 d8 9c cd ab ff ff 8e  a3 f0 30 3e 36 ce f4 20   A······ ··0>6··
0010  75 73 65 72 3a 20 44 6f  6e 42 6f 73 63 6f 3b 20   user: Do nBosco;
0020  70 77 64 3a 20 56 61 6c  73 61 20 2c 01 bb e0      pwd: Val sa ,···
```

Data (data.data), 30 bytes Packets: 23 · Displayed: 23 (100.0%) Profile: Default

Sending IPv6 Packets



Sending IPv6 Packets

1. From the Terminal go to the **ipv6** directory

```
cd ../ipv6
```

2. ⚠ Put the **nRF52840** in **bootloader mode** (blinking red light)

3. Program the firmware

```
make client.dfu-upload
```

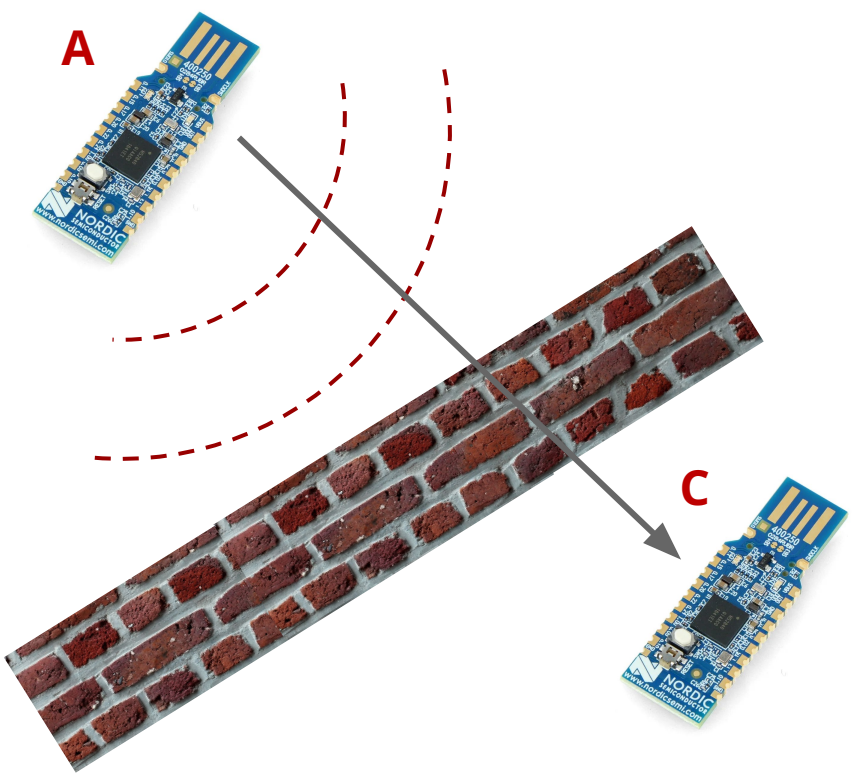
4. Attach to the serial output

```
make login
```

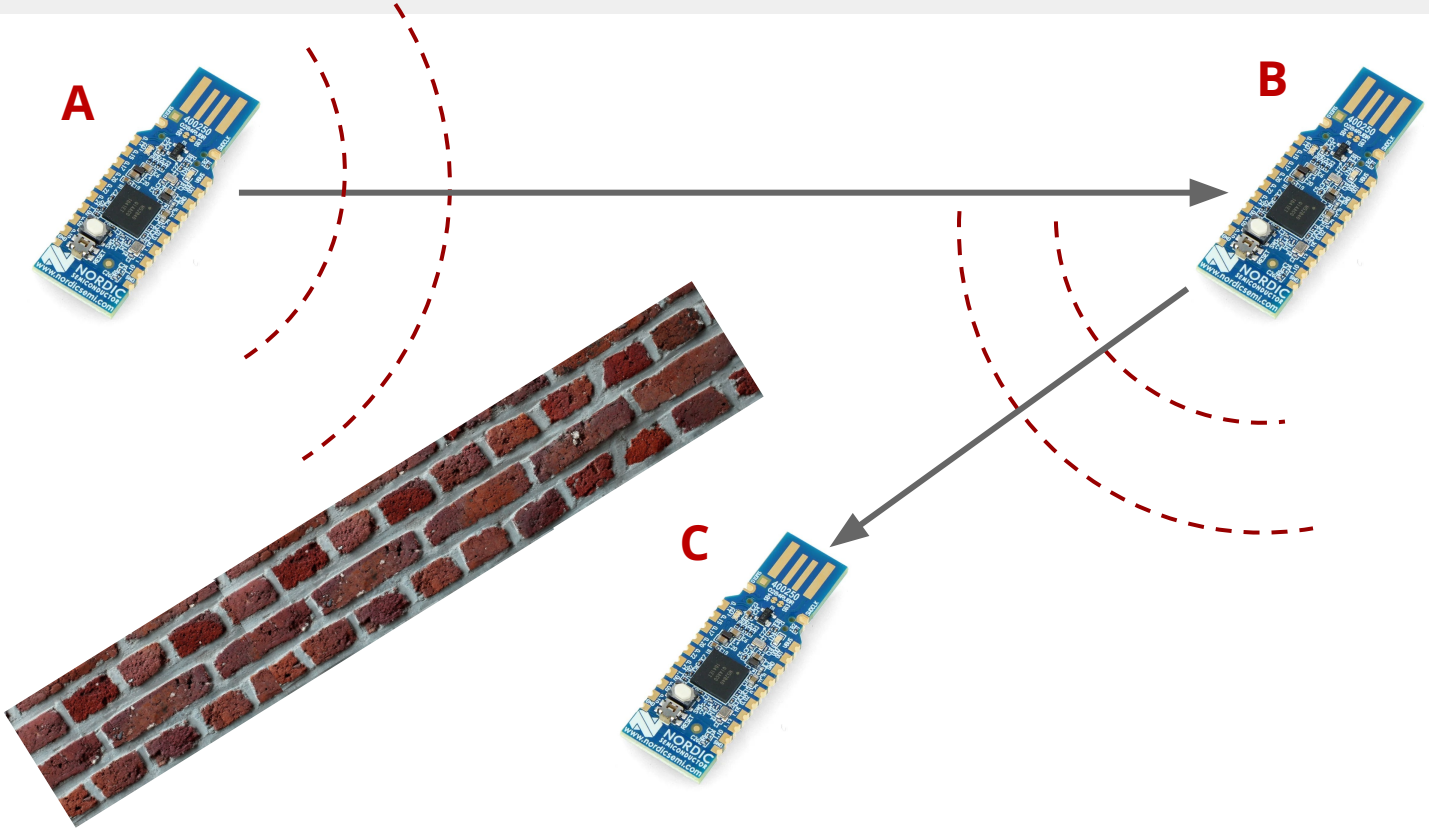
Sending IPv6 Packets

```
[INFO: App        ] Cannot send packet as root node is not reachable yet.
[INFO: App        ] Cannot send packet as root node is not reachable yet.

[INFO: App        ] Sending request to fd00::f6ce:36ae:2430:52fa
[INFO: App        ] Request has data 'hello 0'
[INFO: App        ] Received response from fd00::f6ce:36ae:2430:52fa
[INFO: App        ] Response has data 'hello 0'

[INFO: App        ] Sending request to fd00::f6ce:36ae:2430:52fa
[INFO: App        ] Request has data 'hello 1'
[INFO: App        ] Received response from fd00::f6ce:36ae:2430:52fa
[INFO: App        ] Response has data 'hello 1'
```

Routing (RPL)



Routing (RPL)



Routing (RPL)

1. From the Terminal go to the **ipv6** directory

```
cd ../shell
```

2. ⚠ Put the **nRF52840** in **bootloader mode** (blinking red light)

3. Program the firmware

```
make node.dfu-upload
```

4. Attach to the serial output

```
make login
```

Routing (RPL)

4. To show all known IPv6 addresses

```
ip-addr
```

5. To show all IPv6 neighbors

```
ip-nbr
```

6. To show all RPL routes

```
routes
```

7. To show all RPL neighbors

```
rpl-nbr
```

Routing (RPL)

```
routes

Default route:
-- None

Routing links (2 in total):
-- fd00::f6ce:36de:2f45:8c9e  (DODAG root)
(lifetime: infinite)
-- fd00::f6ce:36ae:2430:52fa  to fd00::f6ce:36de:2f45:8c9e
(lifetime: 1800 seconds)
```

Simulating IPv6

1. From the Terminal go to the **tools/cooja** directory

```
cd ../../tools/cooja
```

2. Run the Cooja simulator using:

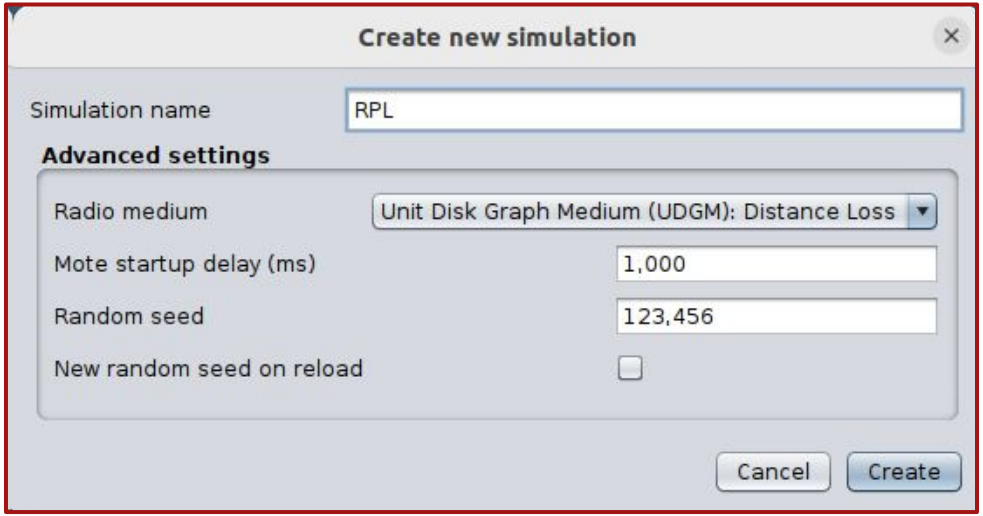
```
./gradlew run
```

Simulating IPv6

3. Create a New Simulation:

`File > New Simulation`

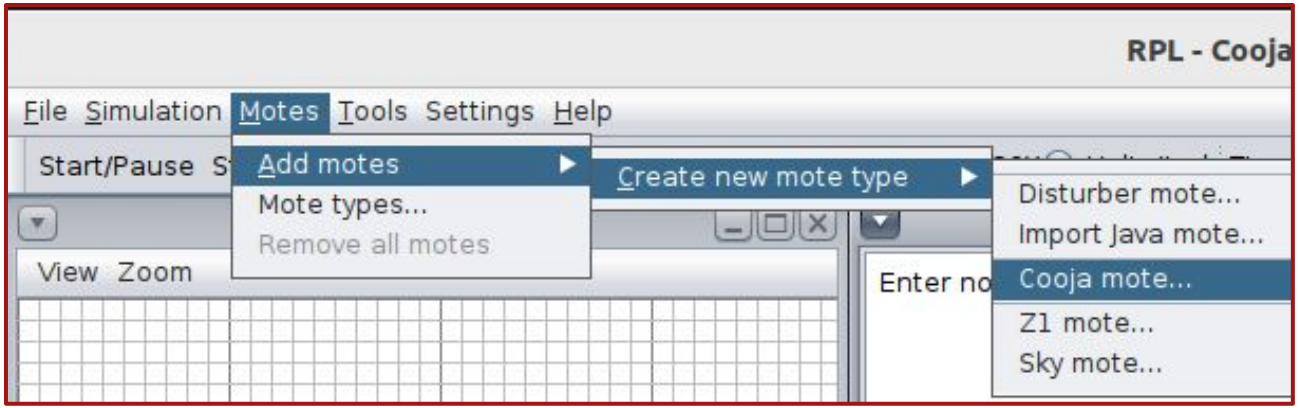
2. Call the Simulation “RPL”, then click “**Create**”



Simulating IPv6

3 Let's add a new Mote:

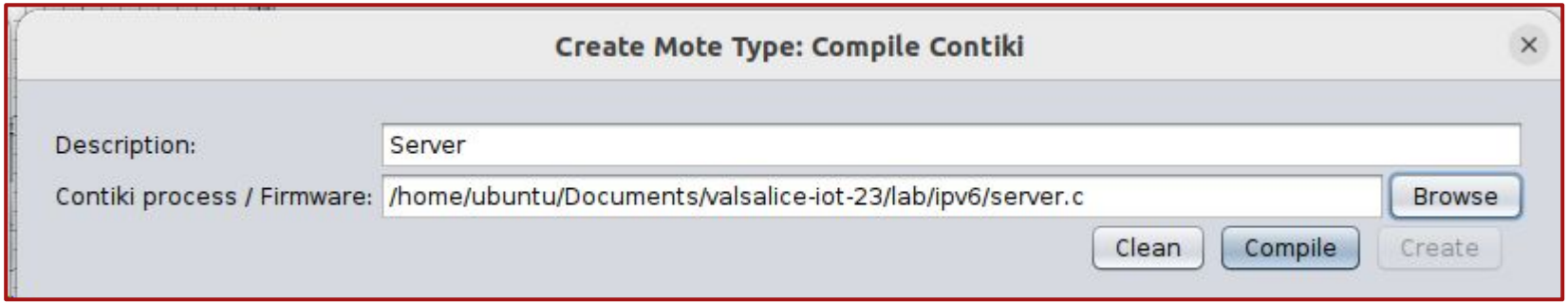
Motes > Add motes > Create new > Cooja mote



Simulating IPv6

4. Put "**Server**" in the description
5. Use the firmware under (you can also use "Browse"):

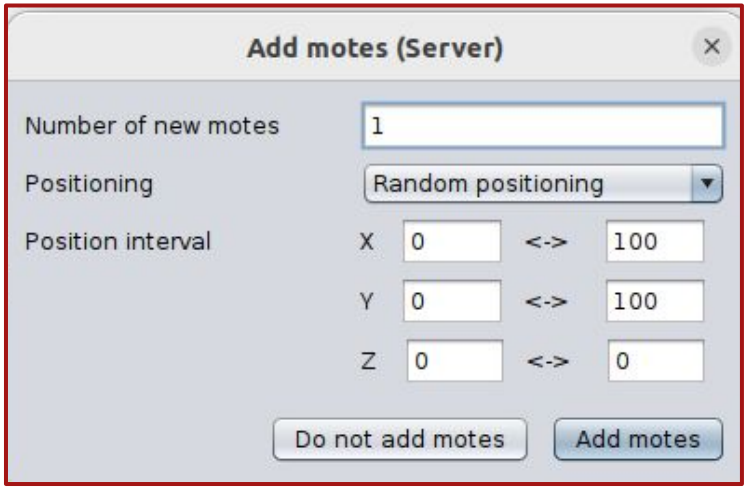
```
/home/ubuntu/Documents/valsalice-iot-23/lab/ipv6/server.c
```



6. Click "**Compile**"
7. Click "**Create**"

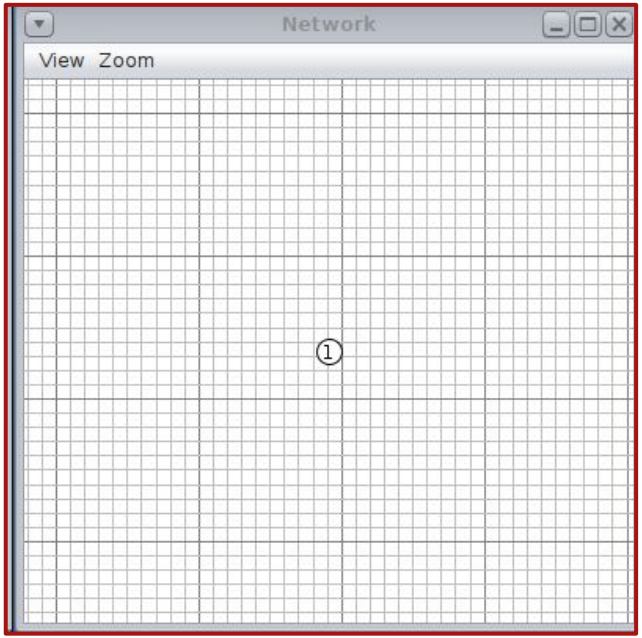
Simulating IPv6

8 Put “**1**” in the “Number of new motes” field



9 Click “**Add motes**”

10 You should get this ——————>



Simulating IPv6

11. Create **ONE** new “Client” mote:

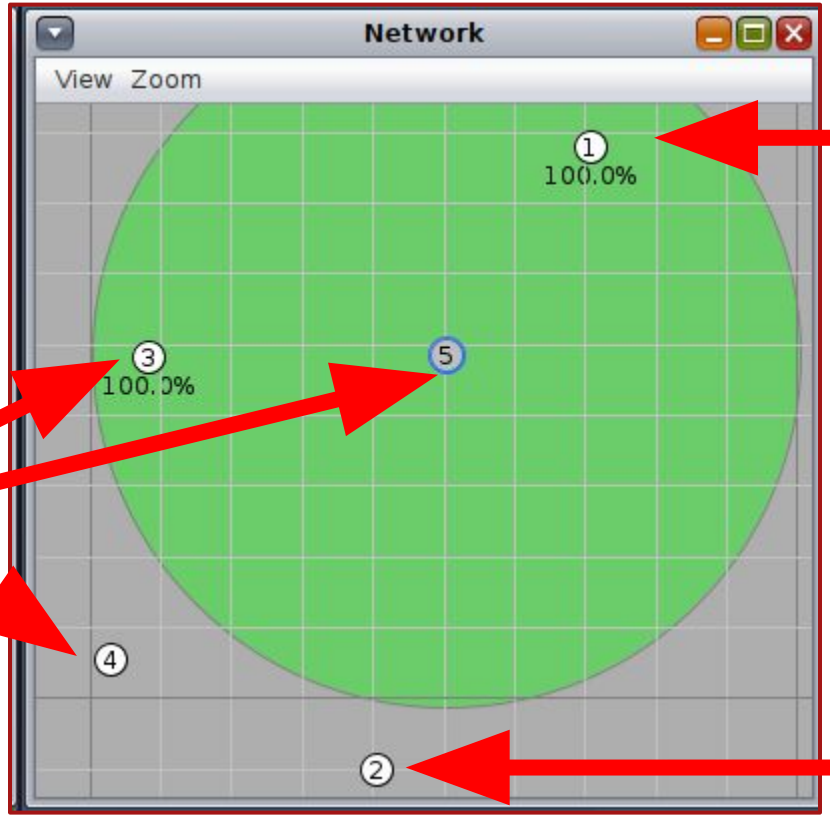
    `/home/ubuntu/Documents/valsalice-iot-23/lab/ipv6/client.c`

12. Create **THREE** new “Node” motes:

    `/home/ubuntu/Documents/valsalice-iot-23/lab/ipv6/node.c`

Simulating IPv6

Target topology:



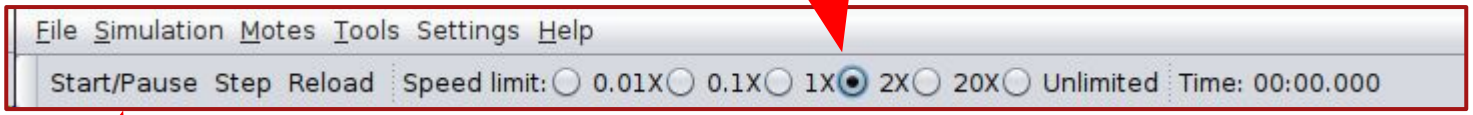
Server

Node

Client

Simulating IPv6

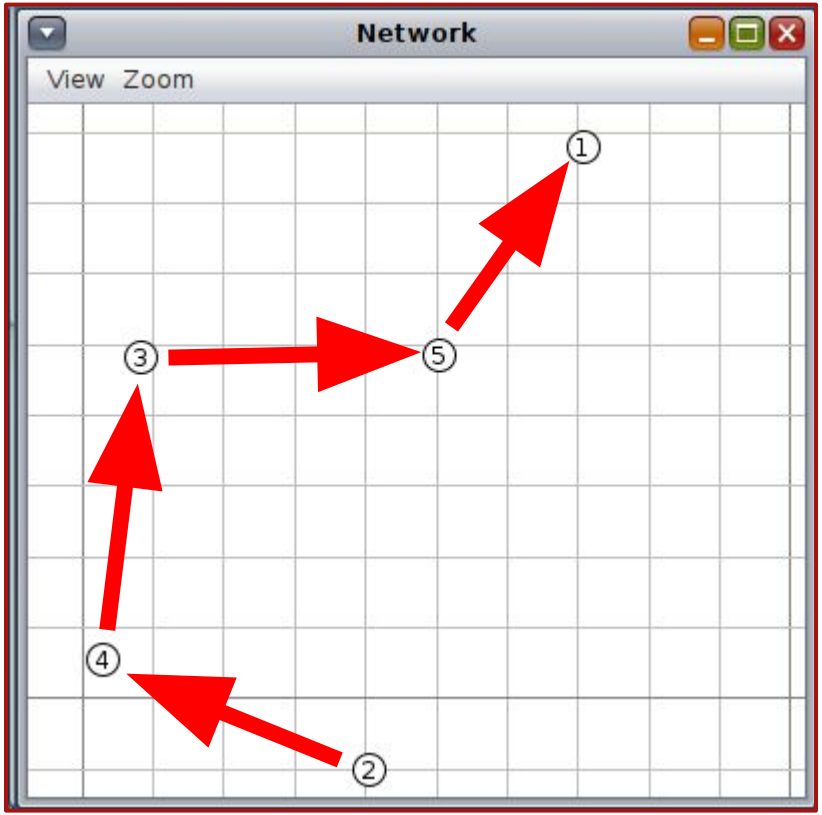
13 Set the simulation speed to **2X**



14 **Start** the simulation

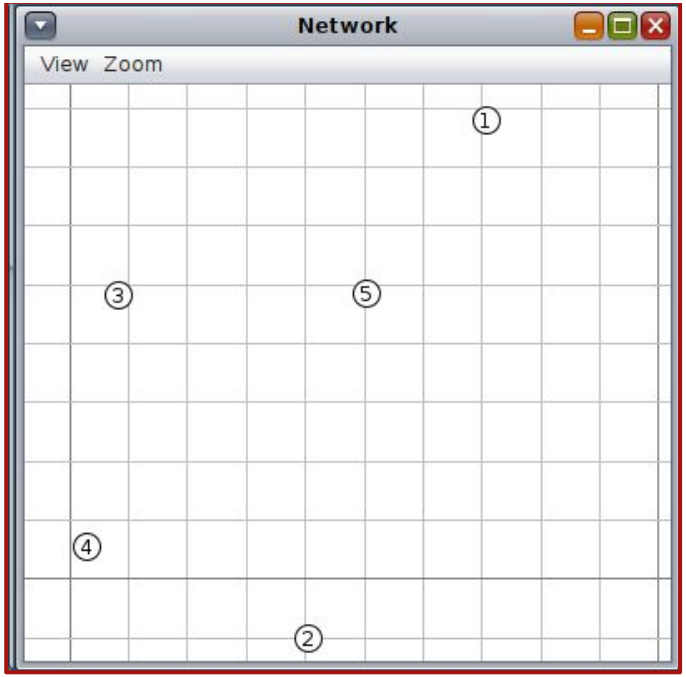
Simulating IPv6

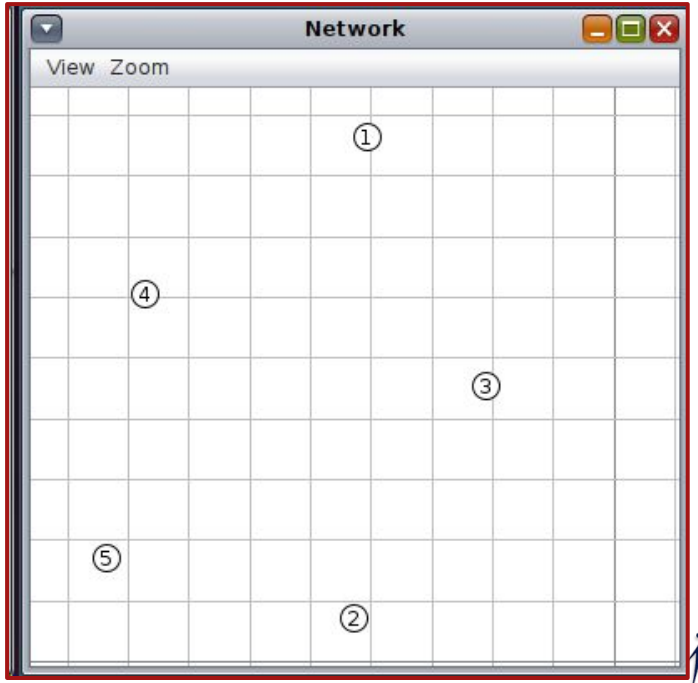
Packet Route:



Simulating IPv6

Experiment with different topologies:





End of Lab

Thank You!