NetSim®

Accelerate Network R & D

Internet of Things (IoT) and Wireless Sensor Networks (WSN)

A Network Simulation & Emulation Software

By

TETCOS LLP

The information contained in this document represents the current view of TETCOS LLP on the issues discussed as of the date of publication. Because TETCOS LLP must respond to changing market conditions, it should not be interpreted to be a commitment on the part of TETCOS LLP, and TETCOS LLP cannot guarantee the accuracy of any information presented after the date of publication.

This manual is for informational purposes only.

The publisher has taken care of the preparation of this document but makes no expressed or implied warranty of any kind and assumes no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information contained herein.

Warning! DO NOT COPY

Contact us at

TETCOS LLP

214, 39th A Cross, 7th Main, 5th Block Jayanagar,

Bangalore - 560 041, Karnataka, INDIA.

Phone: +91 80 26630624

E-Mail: sales@tetcos.com

Visit: www.tetcos.com

Contents

1 Introduction

Internet of Things (IoT) is a network of devices that connect to and communicate via the internet or other communication networks. These devices collect, exchange, and act on data, often without human intervention. It enables enhanced interactivity and automation across various domains, from smart homes to industrial applications. A typical IOT deployment consists of:

- Embedded devices / sensors.
- Communication over an IP network (between the devices and to/from cloud servers).
- Cloud services, Big Data, Analytics / Machine learning on the cloud.

In the context of NetSim, the sensors are abstract, which means that they could be any kind of sensor or embedded device. These sensors are assumed to sense some physical property or random field such as temperature, pressure etc. After sensing, the sensors transmit the sensed data in the form of "IP Packets" of user configurable size and interpacket arrival times. NetSim simulates the transmission of these IP packets over an IoT network and does not focus on either the actual "application payload (or the sensed data)" being sent, or the data storage and analytics of this payload.

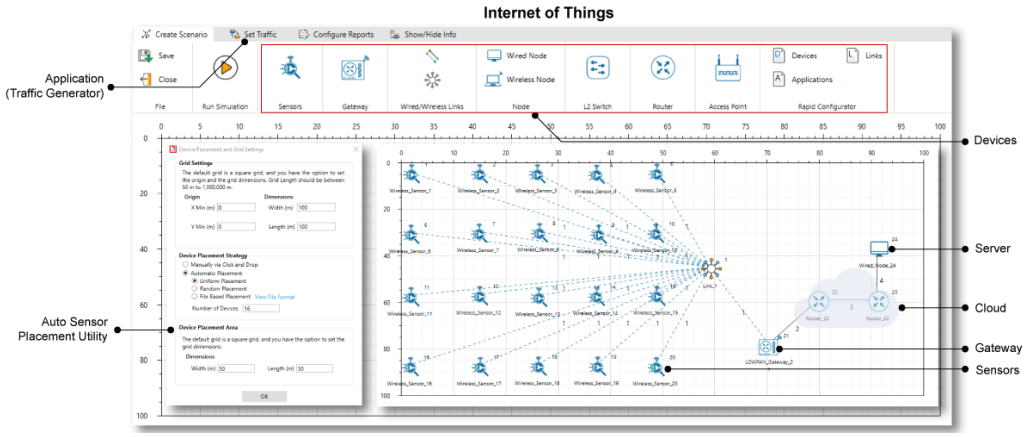


Figure 1-1: *A typical IOT scenario in NetSim.*

NetSim's Internet of Things (IoT) and Wireless Sensor Network (WSN) library stack comprises:

- Application Layer: Sensor App as well as applications such as Voice, Video, CBR etc.
- Transport Layer: UDP
- Network layer: AODV and RPL
- MAC and PHY layers: 802.15.4 Zigbee

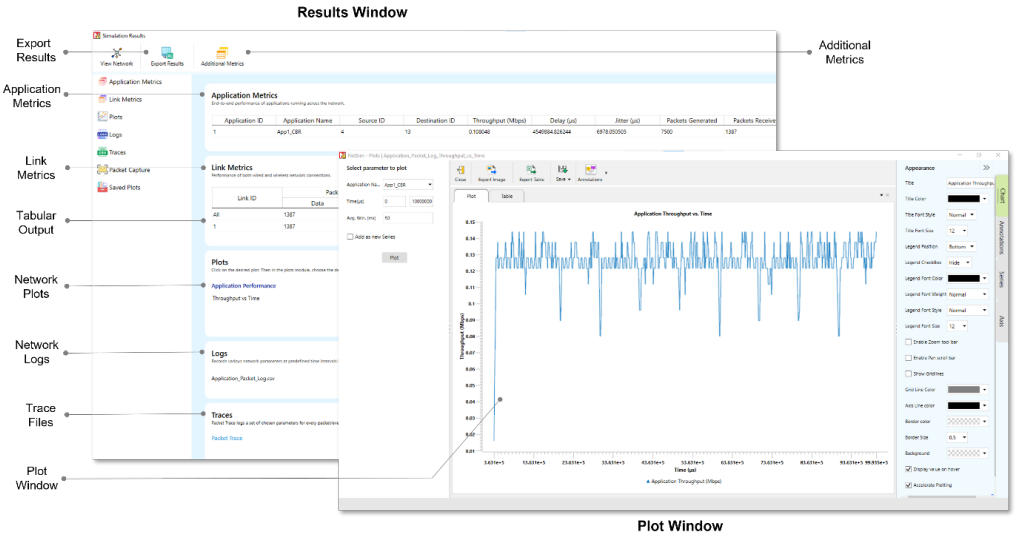


Figure 1-2: *The Result dashboard and Plot window shown in NetSim after completion of simulation.*

NetSim models IoT as a WSN that connects to an Internetwork through a LowPAN Gateway. The LowPAN Gateway uses two interfaces: a Zigbee (802.15.4) interface and a WAN Interface. The WSN sends data to a SinkNode. The Zigbee interface allows wireless connectivity to the WSN while the WAN interface connects to the external Internetwork.

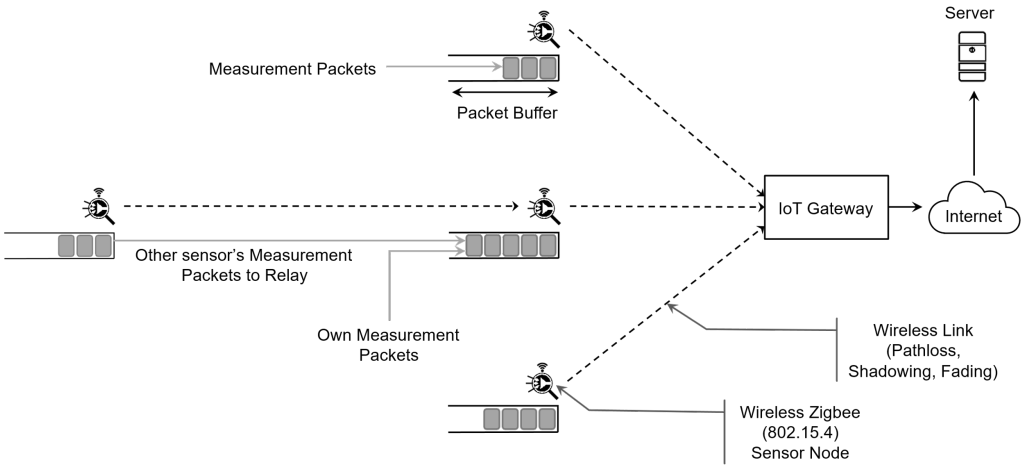


Figure 1-3: *A typical application scenario that can be modeled in NetSim. Sensors can generate (measurement) packets that get queued in its packet buffer. These are then transmitted – directly or via hops – over a wireless link to a gateway that then forwards the packet via the internet to a server. Wireless links support various propagation models and ad hoc routing is supported for multi-hop communication. The MAC/PHY layer protocol supported is 802.15.4.*

IEEE 802.15.4 uses either Beacon Enabled or Disabled Mode for packet transmission. In Beacon Enabled Mode, nodes use slotted CSMA/CA algorithm for transmitting packets else they use Unslotted CSMA/CA.

2 Simulation GUI

In the Main menu select New Simulation → Wireless Sensor Networks as shown in Figure 2-1.

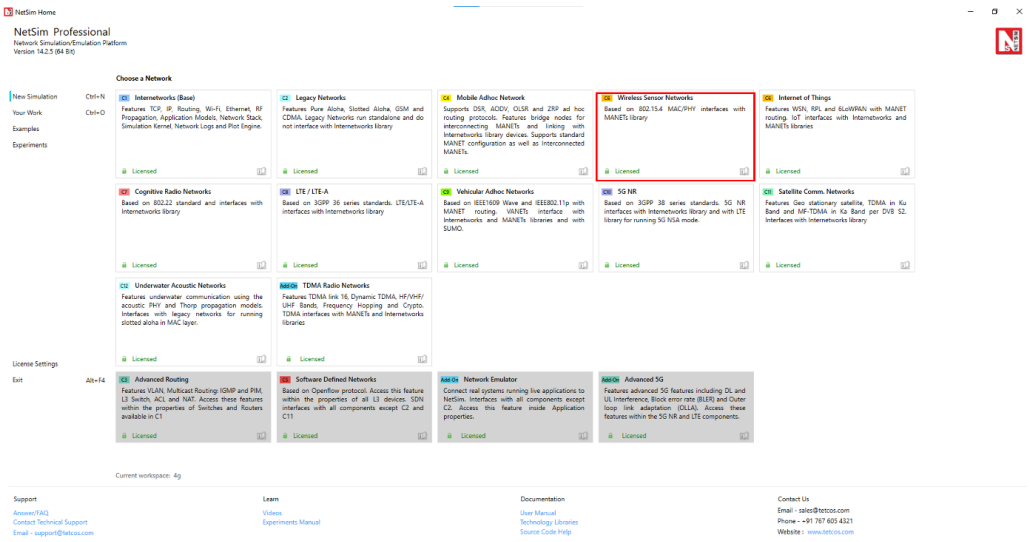


Figure 2-1: *NetSim Home Screen.*

2.1 Create Scenario

2.1.1 Fast Configuration

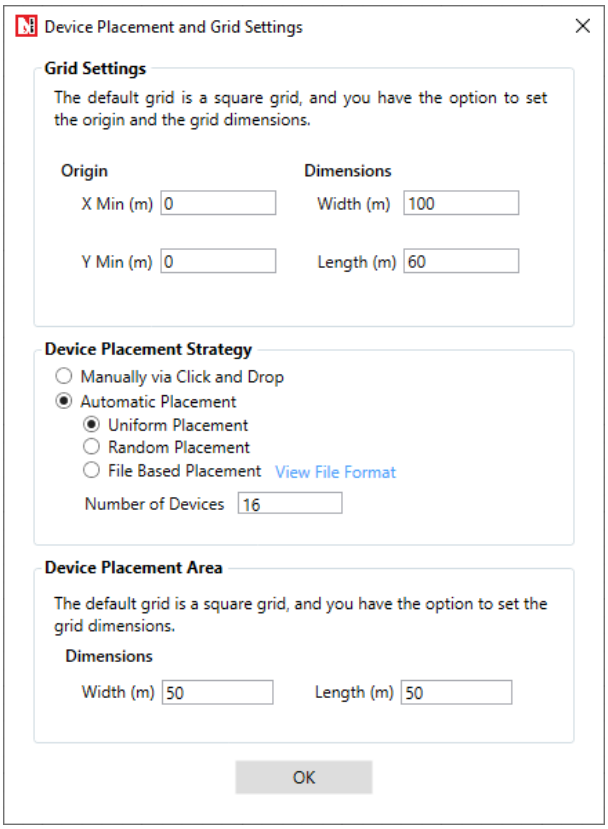


Figure 2-2: *Fast Configuration window.*

Fast Config window allows users to define device placement strategies and conveniently model large network scenarios especially in networks such as MANET, WSN and IoT. The parameters associated with the Fast Config Window are explained below:

(i) Grid Origin: The 'Grid Origin' refers to the intersection point of the system's axes. NetSim supports any (X, Y) setting for the origin and not just (0, 0)

(ii) Grid Dimension: The width parameter represents the maximum along X from the origin and the height parameter represents the maximum along Y from the origin

(iii) Device placement area: The "Device Placement Area" allows users to specify the width and length of the area where devices are used when using the auto placement utility. This area must be less than or equal to the "Grid Area".

Device Placement – Automatic Placement:

1. Uniform Placement: Devices will be placed uniformly with equal gap between the devices in area as specified inside length. This requires users to specify the number of devices as square number. For example, 1, 4, 9, 16 etc.
2. Random Placement: Devices will be placed randomly in the grid environment within the area as specified inside length.
3. File Based Placement: To place devices in user defined locations file-based placement option can be used. The file has the following general format:

```
<DEVICE NAME>,<DEVICE TYPE>,<X COORDINATE>,<Y COORDINATE>
```

Where, DEVICE_NAME is any name that will be assigned to the device. And DEVICE_TYPE is the unique Device Identifier specific to each type of device in NetSim.

The following table provides a list of all possible devices in MANET, WSN, UWAN and IOT Networks that support the Fast Configuration along with their respective device types:

Table 2-1: *Fast Configuration window supports different networks.*

NETWORK	DEVICE TYPE
MANET	Wireless Node Omni Antenna | Wireless Node Sector Antenna | Wired Bridge Node | Wireless Bridge Node | Wired Node | Router | L2 Switch
WSN	Sensors | Sink node
UWAN	Under Water node
IOT	IoT Sensors | Gateway | Wired Node | IoT Router | Access Point

NOTE: For more details about File Based Placement, refer 3.6.

4. Number of Devices: It is the total number of devices that are to be placed in the grid environment. It should be a square number in case of Uniform placement.
5. Manually Via Click and Drop: Selecting this option will load a grid environment with an ad hoc link where users can add devices manually by clicking and dropping the devices as required.

2.1.2 Wireless Sensor Networks

The devices that are involved in WSN are:

Wireless Sensor: In general, sensors monitor and record the physical conditions of the environment which is then sent to a central location (Sink node) where the data is collated and analyzed for further action. Sensors in NetSim are abstract in terms of what they sense, and NetSim focuses on the network communication aspects after sensing is performed.

- WSN Sink (in WSN): Sink node is the principal controller in WSN. All sensors send their data to this sink node. In NetSim, users can drop only one sink node in a WSN.

- Ad-hoc Link: Ad hoc link depicts a decentralized type of wireless network. The network is ad hoc because it does not rely on any pre-existing infrastructure, such as routers in wired networks or access points in managed wireless networks. In NetSim, Ad hoc links are used to connect the Sensors and the Sink node. Ad hoc links are used here for a visual representation of connection of all the devices in an Ad hoc basis.

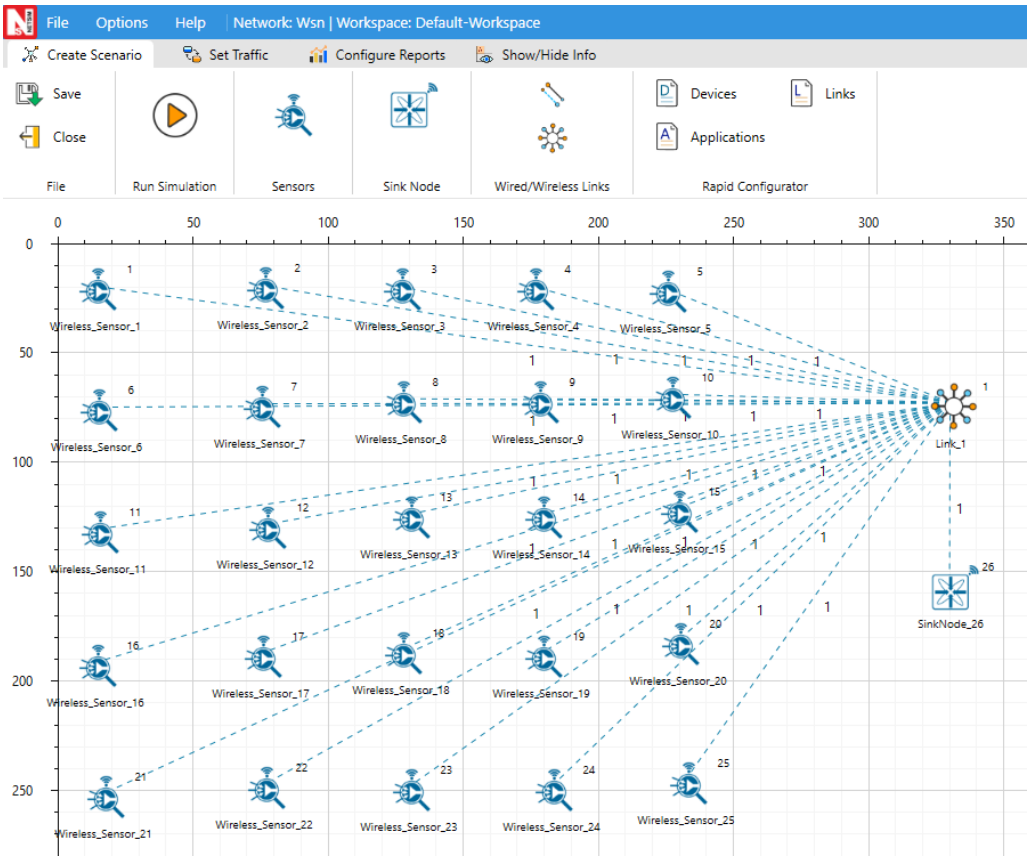


Figure 2-3: *Network layout of a WSN. The sensors communicate with each other and to the sink via an "Adhoc link".*

NOTE: While designing a network, by default an ad hoc link will be present in the scenario. Click sensor nodes and sink nodes present in the ribbon/toolbar and drop them inside the grid. If the auto-connect option in the status bar is turned ON, these devices will be automatically connected to the ad hoc link. Refer section 3.2.3 of User Manual to know more about Auto Connection.

2.1.3 Internet of Things

The devices that are involved in IoT are:

IoT Sensor: In general, sensors monitor and record the physical conditions of the environment which is then sent to a central location (Lowpan Gateway) where the data is collated and analyzed for further action. Sensors in NetSim are abstract in terms of what they sense, and NetSim focuses on the network communication aspects after sensing is performed.

LoWPAN Gateway (in IoT): LoWPAN is an acronym of Low power Wireless Personal Area Networks. The LoWPAN IoT gateway functions as a border router in a LoWPAN network, connecting a wireless IPv6 network to the Internet. The wired portion of the network in IoT runs IPv4 whereas the wireless portion runs IPv6. The IPv6 routing protocols supported are AODV and RPL.

Ad-hoc Link: Ad hoc link depicts a decentralized type of wireless network. The network is ad hoc because it does not rely on any pre-existing infrastructure, such as routers in wired networks or access

points in managed wireless networks. In NetSim IoT, Ad hoc links are used to connect the IoT Sensors and the LowPAN Gateway. Ad hoc links are used here for a visual representation of connection of all the devices in an Ad hoc basis.

Users can also add routers and nodes as shown below. Routers can be connected to the LoWPAN-Gateway and nodes/switches can be connected to routers using wired/wireless links.

2.1.4 Differences between IoT and WSN in NetSim

Table 2-2: *Differences between IoT and WSN in NetSim.*

WSN	IoT
WSN is a network of sensors and a sink node.	IOT has a gateway which can be used to connect to internetworks (having routers, switches, APs etc.).
WSN runs IPv4 and features a sink (not a gateway).	IOT runs IPv6 in the sensor network (802.15.4 MAC/PHY) and IPv4 on the inter-network portion.
Routing protocols in NetSim WSN include, DSR, AODV, OLSR, and ZRP.	Routing protocols in NetSim IoT include, AODV and RPL.

NOTE: Refer MANET Technology library for working of AODV, DSR, OLSR and ZRP.

2.1.5 Device Attributes

GENERAL PROPERTIES

Right click on any sensor and open properties as new window. The general properties of the sensor are:

- **Device name** – It is the name of sensor which is editable and will reflect in the GUI before and after simulation.
- **X and Y** – These are the coordinates of a sensor.
- **Z co-ordinate** – By default this will be zero and is reserved for future use.

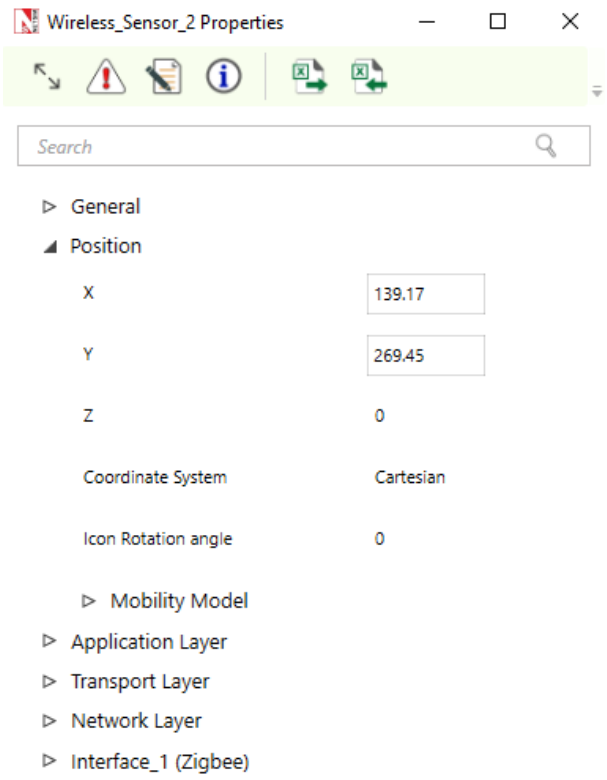


Figure 2-4: *General Properties window for Sensor.*

Interface count is 1 since sensors share the wireless Multipoint-to-Multipoint medium.

- **Mobility Models:** Mobility models can be used to model movement of sensors. The mobility models provided in NetSim are:
  - **Random Walk Mobility model:** It is a simple mobility model based on random directions and speeds.
  - **Random Waypoint Mobility Model:** It includes pause time between changes in direction and/or speed.
  - **Group mobility:** It is a model which describes the behavior of sensors as they move together i.e., the sensors having common group id will move together.
  - **Pedestrian Mobility Model:** This model is applicable to each node (local parameter), and the configuration parameters are:
    - **Pedestrian Max Speed (m/s)** (Range: 0.0 to 10.0. Default: 3.0)
    - **Pedestrian Min Speed (m/s)** (Range: 0.0 to 10.0. Default: 1.0)
    - **Pedestrian stop probability** (Range: 0 to 1)
    - **Pedestrian stop duration (s)** (Range: 1 to 10000)

In this model it is assumed that the pedestrian stops at traffic lights. The stop probability represents the probability of encountering a traffic light. This is checked for every calculation interval. Once stopped, the pedestrian waits for a duration equal to stop duration for the light to turn green. A new direction is chosen randomly after every stop with θ (angle between new direction and current direction) taking values of 0, 90, 180, 270. These θ values represent the pedestrian continuing in the same direction, taking a left, taking a U turn and taking a right respectively.

A new speed is chosen randomly after every stop. Min speed $\leq$ Speed $\leq$ Max speed.

The maximum number of stops and starts is 10.

- **File Based mobility:** In File Based Mobility, users can write their own custom mobility models and define the movement of the mobile users. The name of the trace file generated should be kept as mobility.csv and it should be in the NetSim Mobility File format.

APPLICATION PROPERTIES – Transport Protocol, by default set to UDP. To run with TCP, users have to select TCP protocol from the drop down.

NETWORK LAYER – NetSim WSN, supports the following MANET routing protocols.

1. **DSR (Dynamic source routing):** Note that in wireless sensor networks, by default Link Layer Ack is enabled. If Network Layer ack is enabled users must set DSR ACK in addition to Zigbee ACK in MAC layer.
2. **AODV (Ad-hoc on-demand distance vector routing):**
3. **ZRP (Zone routing protocol):** For interior routing mechanism NetSim uses OLSR protocol.
4. **OLSR (Optimized link State Routing):** Except zone radius all the parameters are similar to ZRP.

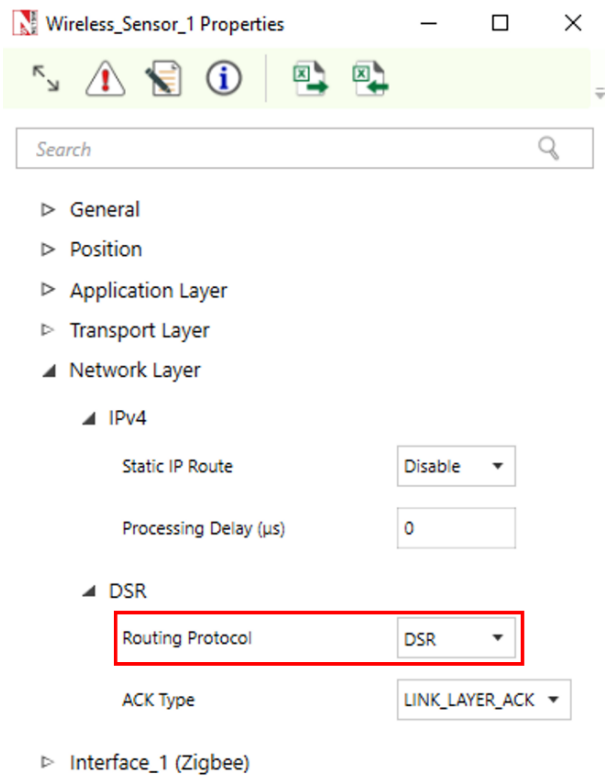


Figure 2-5: *Network Layer Properties Window – DSR Protocol.*

AODV – AODV (Ad Hoc on Demand Distance Vector) is an on-demand routing protocol for wireless networks that uses traditional routing tables to store routing information. AODV uses timers at each node and expires the routing table entry after the route is not used for a certain time.

Some of the features implemented in NetSim are:

- RREQ, RREP and RERR messages.
- Hello message.
- Interface with other MAC/PHY protocols such as 802.15.4, TDMA / DTDMA.

Hello interval – It describes the interval in which it will discover its neighbor routes.

Refresh interval – It is the duration after which each active node periodically refreshes routes to itself.

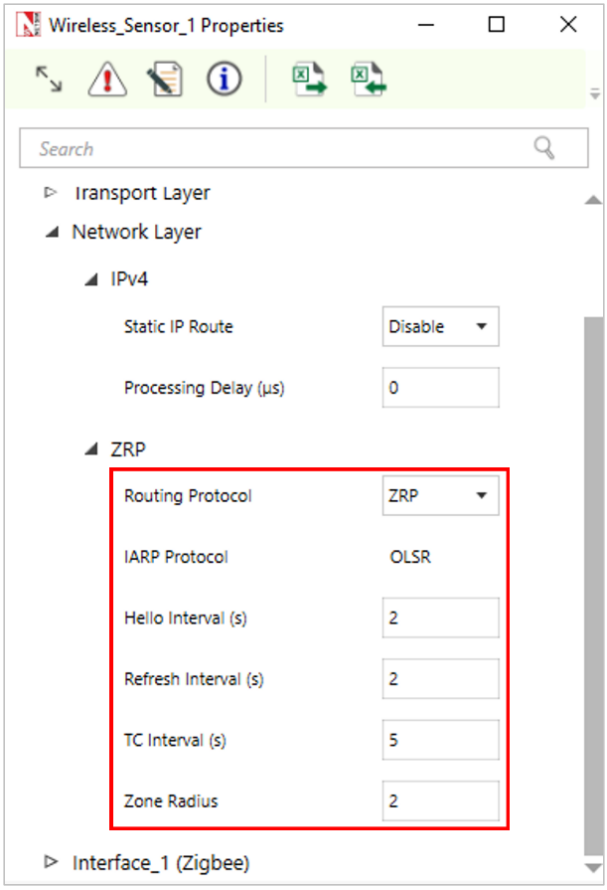


Figure 2-6: *Network Layer Properties Window – ZRP Protocol.*

Topology Control messages – These are the link state signaling done by OLSR. These messages are sent at TC interval every time.

Zone radius – After dividing the network range of the divided network will be based on zone radius. A node's routing zone is defined as a collection of nodes whose minimum distance in hops from the node in question is no greater than a parameter referred to as the zone radius.

DATALINK LAYER

802.15.4 (Zigbee Protocol) runs in MAC layer. In the sink node or pan coordinator properties users can configure the Beacon frames and the Superframe structure.

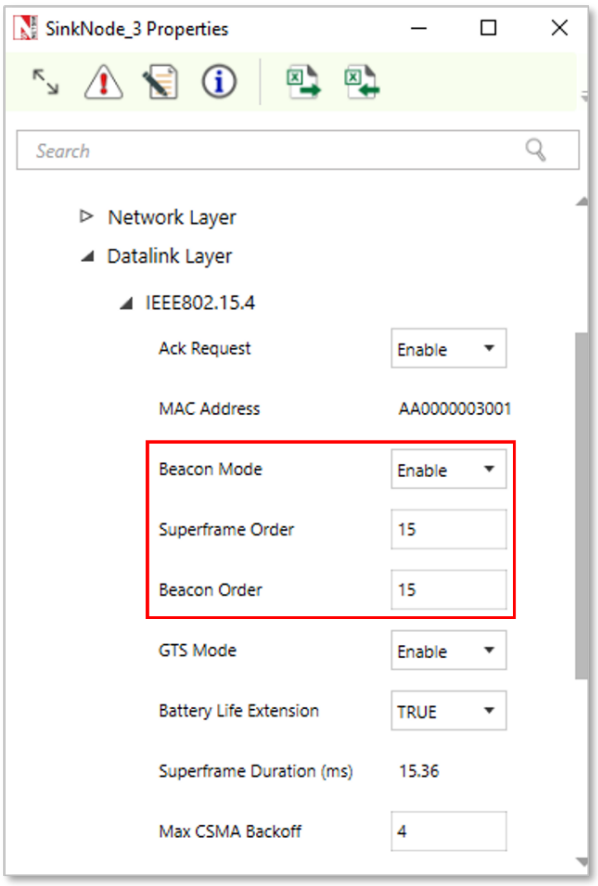


Figure 2-7: *Datalink layer properties window for sinknode.*

Superframe Order – It describes the length of the active portion of the Superframe, which includes the beacon frame. Range is from 0–15.

Beacon Order – Describes the interval at which coordinate shall transmit its beacon frames. Range is from 1–15.

GTS Mode (Guaranteed Time Slot) – If it is enabled it allows a device to operate on the channel within a portion of the super frame that is dedicated (on the PAN) exclusively to the device.

Battery life Extension – This subfield is 1 bit in length and shall be set to one if frames transmitted to the beaconing device.

Superframe Duration – It is divided into 16 equally sized time slots, during which data transmission is allowed. The value of super-frame duration by default is 15.36ms.

Max CSMA Backoff – It is the maximum number of attempts the CSMA-CA algorithm will make before declaring a channel access failure. Having range 0–5.

Minimum CAP length – It is the minimum number of symbols forming the Contention access period. This ensures that MAC commands can still be transferred to devices when GTSs (Guaranteed time slots) are being used.

Max and Min Backoff Exponent – Values of CSMA-CA algorithms have a range of 3–5.

Max Frame Retries – It is the total number of retries after failed attempts.

Unit Backoff Period – It is the number of symbols forming the basic time period used by the CSMA-CA algorithms.

PHYSICAL LAYER

The frequency band used in NetSim WSN simulations is 2.4 GHz, and the bandwidth is 5 MHz. NetSim simulates a single channel ZigBee network and does not support multiple channels.

Data rate – It is the number of bits that are processed per unit of time. The data rate is fixed at 250 kbps per the 802.15.4 standard.

Chip Rate – A chip is a pulse of direct sequence spread spectrum code, so the chip rate is the rate at which the information signal bits are transmitted as pseudo random sequence of chips.

Modulation technique – O-QPSK (Offset quadrature phase shift keying), sometimes called staggered quadrature phase shift keying, is a variant of phase-shift keying modulation using 4 different values of the phase to transmit.

MinLIFSPeriod – It is the minimum long inter-frame spacing period. It is the time difference between short frame and long frame in unacknowledged case and the time difference between short frame and acknowledged case in acknowledgment transmission.

SIFS (Short inter-frame Symbol) – It is generally the time for which receiver waits before sending the CTS (Clear To Send) & acknowledgement packet to sender, and sender waits after receiving CTS and before sending data to receiver. Its main purpose is to avoid any type of collision. Min SIFS period is the minimum number of symbols forming a SIFS period.

Phy SHR duration – It is the duration of the synchronization header (SHR) in symbol for the current PHY.

Phy Symbol per Octet – It is the number of symbols per octet for the current PHY.

Turn Around Time – Transmitter to receiver or receiver to transmitter turnaround time is defined as the shortest time possible at the air interface from the trailing edge of the last chip (of the first symbol) of a transmitted PLCP protocol data unit to the leading edge of the first chip (of the first symbol) of the next received PPDU.

CCA (Clear Channel Assessment) – It is a carrier sensing mechanism in Wireless Networks. The different types of CCA modes available are:

- Carrier Sense Only: It shall report a busy medium only upon the detection of a signal compliant with this standard with the same modulation and spreading characteristics of the PHY that is currently in use by the device. This signal may be above or below the ED threshold.
- Energy Detection: It shall report a busy medium upon detecting signal strength above the ED threshold.
- Carrier Sense with Energy Detection: It shall report a busy medium using a logical combination of detection of a signal with the modulation and spreading characteristics of this standard and Energy above the ED threshold, where the logical operator may be AND or OR.

Receiver sensitivity – It is the minimum magnitude of input signal required to produce a specified output signal having a specified signal-to-noise ratio, or other specified criteria. It is up to the user to determine the desired receiver sensitivity.

Receiver ED threshold – It is intended for use by a network layer as part of channel selection algorithms. It is an estimate of the received signal power within the bandwidth of the channel. No attempt is made to identify or decode signal on the channel. If the received signal power is greater than the ED threshold value, then the channel selection algorithms will return false.

Transmitter Power – It is the signal intensity of the transmitter. The higher the power radiated by the transmitter's antenna, the greater the reliability of the communication system. The connection medium is wireless.

POWER MODEL

- **Power source** – It can be battery or main line. This model in NetSim is used for energy calculations. In case of battery, the following parameters will be considered:
- **Recharging current** – It is the current flow during recharging. Range is from 0–1000mA.
- **Energy harvesting** – It is the process by which energy is derived from external source, captured, and stored. NetSim supports an abstract Energy Harvesting model. A specified amount of energy, calculated from recharging current and voltage specified, is added to the remaining energy of the node periodically to replenish the battery. It can be turned on or off.

- **Initial Energy** – It is the battery energy. Range is from 0.001–3250mAh.
- **Transmitting current** – It is the current for transmitting power. Range is 0–1000mA. Transmit power and transmit current are independent in NetSim. Since the focus of NetSim is packet simulation, the power modeling is abstract. It is left to the user to change the transmit current accordingly, when increasing or decreasing the transmit power, if the user's goal is to study power consumption.
- **Idle mode** – It is the current flow during the idle mode. Range is between 0–1000mA.
- **Voltage** – It is a measure of the energy carried by the charge. Range is from 0–10V.
- **Receiving current** – It is the current required to receive the data, ranging from 0–1000mA.
- **Sleep mode current** – It is the current flowing in sleep mode of battery. Range is from 0–1000mA.

NOTE: The resultant energy metrics and their definitions are provided in the NetSim User Manual in the Outputs section.

The following table shows the properties of sensor in NetSim.

Table 2-3: *MAC and PHY layer properties of sensor.*

Property	Default Setting
Global properties (and default settings)	
Network layer	
Routing protocol	DSR
ACK_Type	LINK_LAYER_ACK
Data link layer	
ACK request	Enable
Max Csma BO	4
Max Backoff Exponent	5
Min Backoff Exponent	3
Max frame retries	3
Local properties (and Default settings)	
Physical layer	
phySHRduration(symbols)	3
Physymbolperoctet	5.3
CCA mode	CARRIER SENSE ONLY
Receiver sensitivity(dbm)	−85
ED threshold (dbm)	−95
Transmitter power(mW)	1
Power	
Power source	Battery
Energy harvesting	ON
Recharging current (mA)	0.4
Initial energy (mAH)	300
Transmitting current (mA)	17
Idle mode current (mA)	3.3
Voltage (v)	3.6
Receiving current (mA)	9.6

Continued on next page

Table 2-3: *MAC and PHY layer properties of sensor (continued).*

Property	Default Setting
Sleep mode current (mA)	0.237

2.2 Set Node, Link and Application Properties

- Users need to connect the sensors and LoWPAN gateway using links.
- Interconnection among other devices is same as in Internetworks.
- LoWPAN gateway can be connected with router using links.
- Click on the appropriate node or link to open its properties in the property panel on the right. Then modify the parameters according to the requirements.
- Routing protocol in Application Layer of routers and all user editable properties in Datalink layer and Physical Layer of Access Point and Wireless Node are Global/Local.
- In Sensor node, Routing protocol in Network Layer and all user editable properties in Datalink layer, Physical Layer and Power are Global/Local.
- **NOTE:**
  - **Global** – Changing properties in one node will automatically reflect in any other nodes in that network.
  - **Local** – Changing properties in one node will not reflect in any other nodes in that network.
- The following are the main properties of sensor node in PHY and Datalink layers as shown in Figure 2-9/Figure 2-10/Figure 2-11.

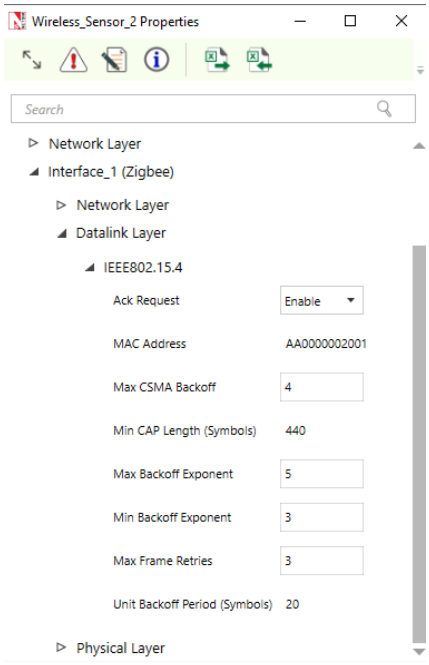


Figure 2-8: *Datalink layer properties window for sensor.*

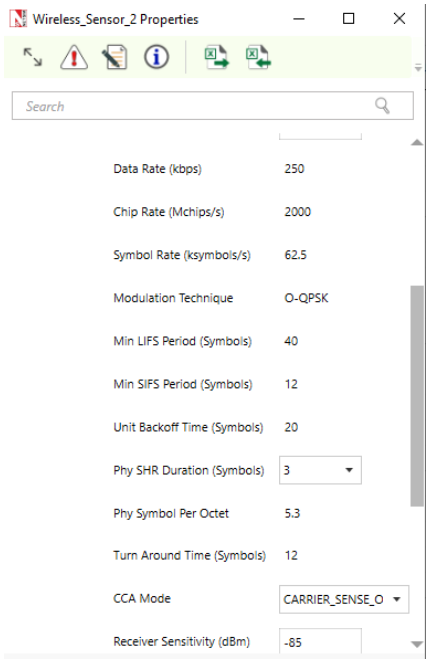


Figure 2-9: *PHY layer properties window for sensor.*

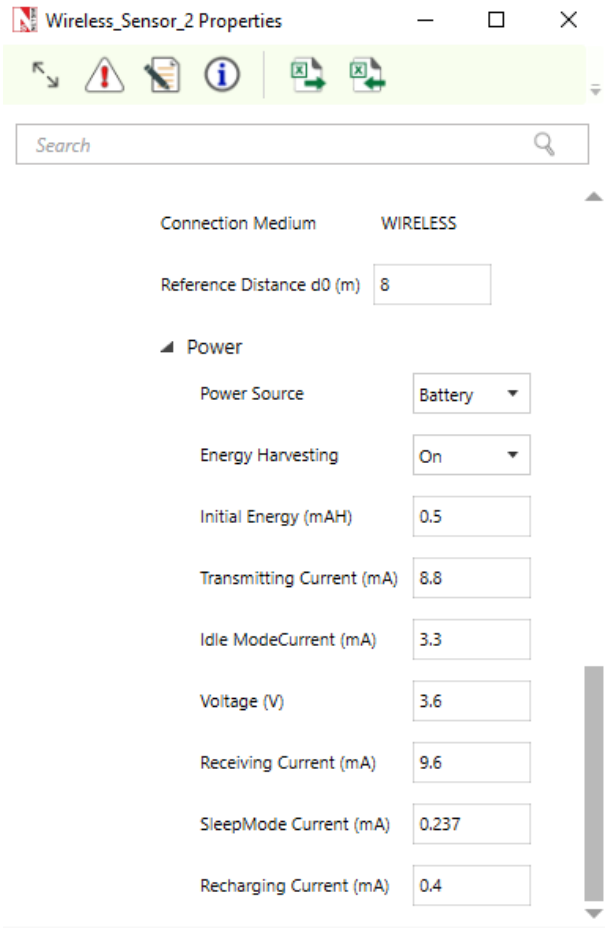


Figure 2-10: *Battery Model for Sensor.*

- Set the values according to requirement and proceed.
- Click on the Set Traffic tab present on the top ribbon and select an application.

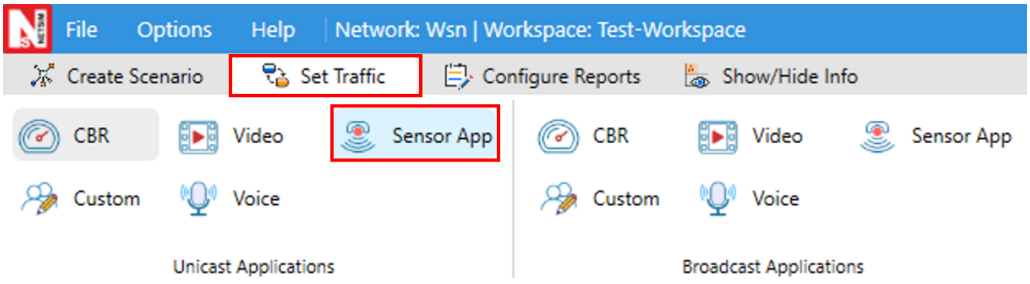


Figure 2-11: *Application icon present on top ribbon.*

- Set the application properties as per the requirements and proceed.

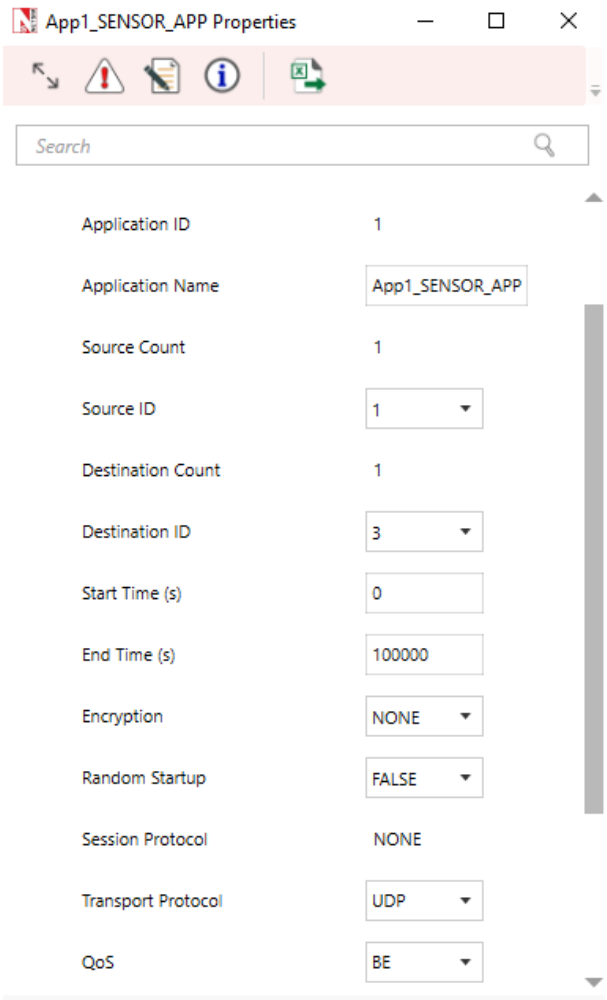


Figure 2-12: *Application Configuration window.*

Detailed information on Application properties is available in section 6 of NetSim User Manual.

2.2.1 Setting Static Routes

In Device Properties ▷ Network layer ▷ Static IP Route, users can set static routes. When static routes are set the dynamic routing protocol entries are overwritten by the static routing entries. Static route configuration is explained in the Internetworks technology library document, Section Configuring Static Routing in NetSim.

Static route option is available for all sensors in WSN.

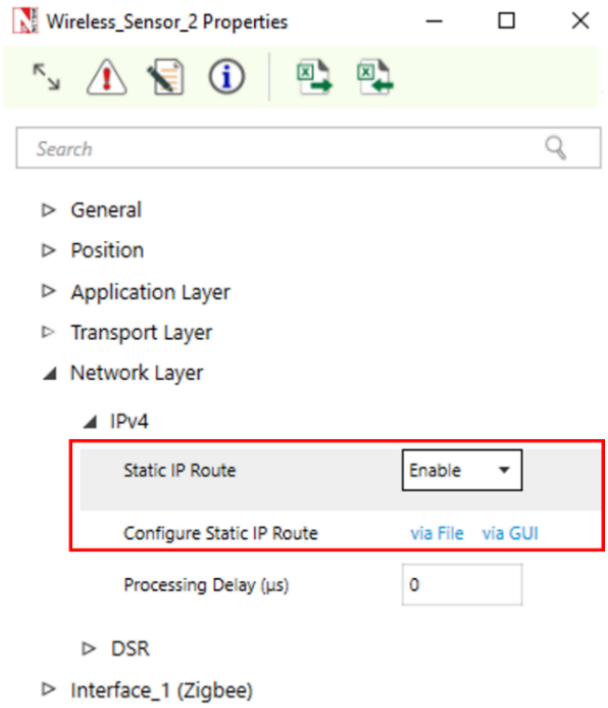


Figure 2-13: *Static Route configuration window.*

The Static routes option is not available in the wireless portion of the IoT network as IoT devices work with IPv6 network addressing. The devices present in the wired portion of the network may have IPv4 addressing. Hence static routes can be configured in the wired section (till the gateway) of an IoT network.

Configure Reports

2.2.2 Enable Packet Trace, Event Trace (Optional)

Check Packet Trace / Event Trace option from the Configure Reports tab. To get detailed help, please refer to sections 8.4 and 8.5 in User Manual.

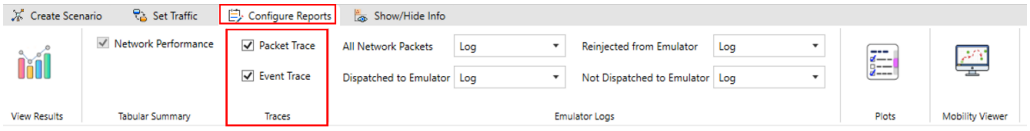


Figure 2-14: *Enable Packet Trace, Event Trace options on top ribbon.*

2.2.3 Enable protocol specific plots

- Click on "Plots" icon in the top ribbon from "Configure Reports" tab, which will then open a right plot panel, containing a list of available plots for the network as chosen by the user.
- Check boxes can be enabled to generate the respective plots. To get detailed help, please refer to section 8.2 in User Manual.

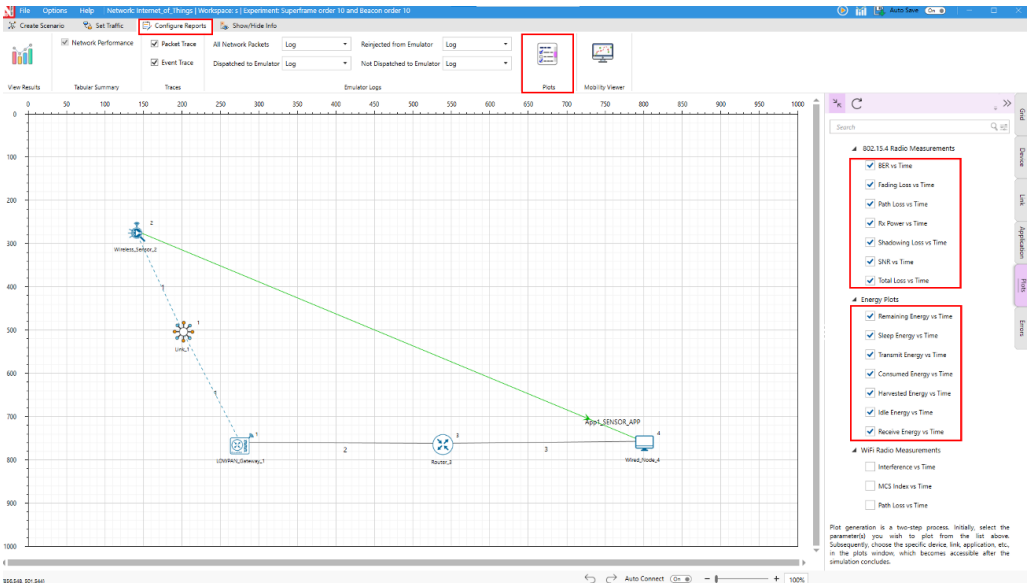


Figure 2-15: *Click on "Configure reports" tab on top. Then click on "plots" icon in the top ribbon. You can then see the list of available plots is in the right-hand side properties panel.*

2.2.4 Enable protocol specific logs

- Users can enable protocol specific log files such as the Radio Measurement log, Energy log etc., by clicking on the Configure Reports tab and selecting Plot icon option present in the toolbar.
- Check boxes can be enabled to generate the respective logs. To get detailed help, please refer to section 8.3 in the User Manual.

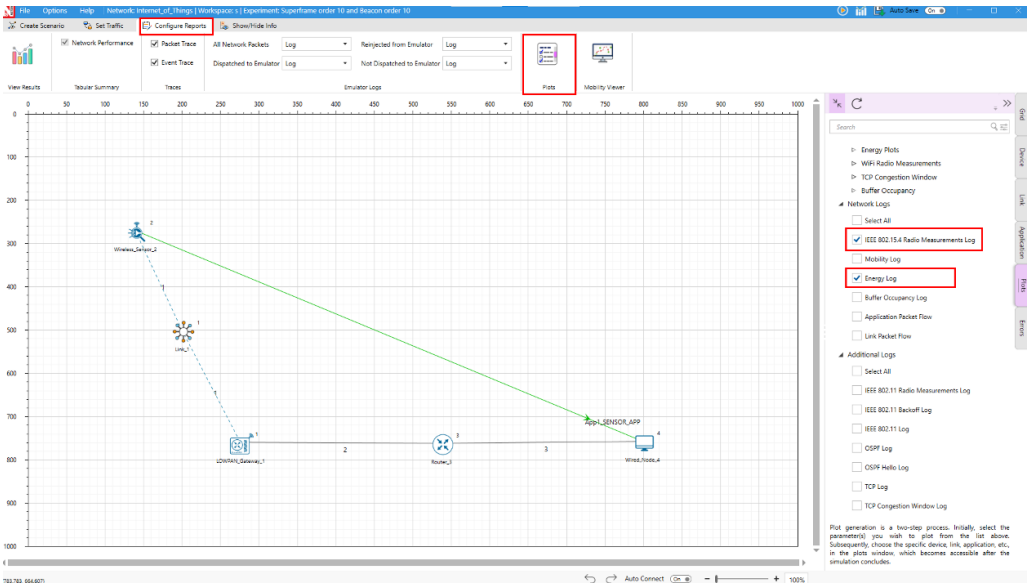


Figure 2-16: *Enabling log files in NetSim GUI.*

2.2.5 GUI Configuration Parameters

Table 2-4: *Data link layer, Physical layer and Network layer properties of Sensor.*

Parameter	Scope	Range	Description
Sensor-Interface(Wireless) – Datalink Layer			
Ack Request	Local	Enable or Disable	The Ack Request setting allows a device to control whether it wants a confirmation (acknowledgment) that its message has been received. Enable Ack Request: When this is turned on, every time the device sends a message, it includes a request for acknowledgment (Ack) in the message's control information. The recipient must send back a confirmation message (an acknowledgment) to let the sender know the message was successfully received. Disable Ack Request: When this is turned off, the device sends messages without asking for confirmation, so the recipient does not send an acknowledgment back.
Mac Address	Fixed		The MAC address is a unique value associated with a network adapter. This is also known as hardware address or physical address. This is a 12-digit hexadecimal number (48 bits in length).
MAC CSMA Backoff	Local	0–5	The maximum number of backoffs the CSMA-CA algorithm will attempt before declaring a channel access failure.
Maximum Backoff Exponent	Local	3–8	The maximum value of the backoff exponent (BE) in the CSMA-CA algorithm.
Minimum Backoff Exponent	Local	3–8	The minimum value of the backoff exponent (BE) in the CSMA-CA algorithm
Max Frame retries	Local	0–7	The maximum number of retries allowed after a transmission failure.
Unit Backoff Period	Fixed	20	The number of symbols forming the basic time period used by the CSMA-CA algorithm
Sinknode-Interface(Wireless) – Datalink Layer			

Continued on next page

Table 2-4: *Data link layer, Physical layer and Network layer properties of Sensor. (continued).*

Parameter	Scope	Range	Description
Beacon Mode	Local	Enable / Disable	Beacon Mode is a setting that affects how a network stays in sync and manages reliable communication: Enable Beacon Mode: When Beacon Mode is turned on, the network sends regular signals (called "beacons") to keep all devices synchronized. This improves reliability because all devices know when they can send and receive messages, making communication more organized. Disable Beacon Mode (Beaconless): When Beacon Mode is turned off, devices communicate using a simpler method called CSMA-CA (Carrier Sense Multiple Access with Collision Avoidance). This approach is lighter and requires less coordination, but it may be less reliable since devices try to avoid message collisions.
Superframe Order	Local	0–15	Superframe Order describes the length of the active portion of the superframe, which includes the beacon frame.
Beacon Order	Local	1–15	Beacon Order, describes the interval at which the coordinator shall transmit its beacon frames.
GTS Mode	Local	Enable/ Disable	GTS Mode (Guaranteed Time Slot Mode) is a setting that decides if a device gets its own reserved time to communicate on the network: Enable GTS Mode: When GTS Mode is enabled, a specific time slot within the network's schedule (superframe) is reserved just for the device. This dedicated time slot means the device can use the channel without interference from others, ensuring reliable and uninterrupted communication. Disable GTS Mode: When GTS Mode is off, the device shares the channel with other devices without a dedicated time slot, which may cause it to wait or experience interruptions.

Continued on next page

Table 2-4: *Data link layer, Physical layer and Network layer properties of Sensor. (continued).*

Parameter	Scope	Range	Description
Battery Life Extension	Local	TRUE/False	The Battery Life Extension (BLE) setting is a single on/off switch (1 bit) that helps save battery by controlling when devices send data after a beacon signal. BLE set to 1 (On): When BLE is turned on, devices must send data to the main device (beaconing device) within a short, specific time after the beacon signal. This short time window, known as the "backoff period," helps the device finish communication quickly, saving battery by reducing how long it stays active. BLE set to 0 (Off): When BLE is off, there is no strict time limit on when the device must start sending data after the beacon. This uses more battery because the device may stay active longer, but it allows more flexibility in timing.
Superframe Duration (ms)	Local	15.36	Superframe Duration is divided into 16 equally sized time slots, during which data transmission is allowed. The value of super-frame duration by default is 15.36ms.
Interface Wireless – Physical Layer			
Data Rate (Kbps)	Fixed		Data rate or bit rate is the number of bits that are conveyed or processed per unit of time.
Chip Rate	Fixed		The chip rate of a code is the number of pulses per second (chips per second) at which the code is transmitted (or received). The chip rate is larger than the symbol rate, meaning that one symbol is represented by multiple chips.
Symbol Rate	Fixed		In digital communications, symbol rate (also known as baud or modulation rate) is the number of symbol changes (waveform changes or signaling events) made to the transmission medium per second using a digitally modulated signal or a line code.
Modulation Technique	Fixed		O-QPSK (Offset quadrature phase shift keying) sometimes called as staggered quadrature phase shift keying is a variant of phase-shift keying modulation using 4 different values of the phase to transmit.
Min LIFS Period	Fixed		The minimum number of symbols forming a LIFS (Long Inter Frame Spacing) period.
Min SIFS Period	Fixed		The minimum number of symbols forming a SIFS (Short Inter Frame Spacing) period.
Phy SHR duration	Local	3,7,10,40	Phy SHR duration is the duration of the synchronization header (SHR) in symbol for the current PHY

Continued on next page

Table 2-4: *Data link layer, Physical layer and Network layer properties of Sensor. (continued).*

Parameter	Scope	Range	Description
Phy Symbol per Octet	Fixed		Phy Symbol per Octet is number of symbols per octet for the current PHY
Turn Around Time	Fixed		Turnaround Time is the minimum wait time needed for a device to switch from sending to receiving data (or vice versa) in a wireless network.
CCA Mode	Local		CCA (Clear Channel Assessment) is a carrier sensing mechanism in Wireless Networks. The different types of CCA modes available are: Carrier Sense Only: It shall report a busy medium only upon the detection of a signal compliant with this standard with the same modulation and spreading characteristics of the PHY that is currently in use by the device. This signal may be above or below the ED threshold Energy Detection: It shall report a busy medium upon detecting signal strength above the ED threshold. Carrier Sense with Energy Detection: It shall report a busy medium using a logical combination of detection of a signal with the modulation and spreading characteristics of this standard and Energy above the ED threshold, where the logical operator may be AND or OR.
Receiver sensitivity	Local	−999 to 0	Receiver sensitivity is the minimum magnitude of input signal required to produce a specified output signal having a specified signal-to-noise ratio, or other specified criteria. It is up to the user to determine the desired receiver sensitivity.
ED threshold(dBm)	Local	−999 to 0	The receiver ED threshold is intended for use by a network layer as part of a channel selection algorithm. It is an estimate of the received signal power within the bandwidth of the channel. No attempt is made to identify or decode signals on the channel. If the receive signal power is greater than the ED threshold value then the channel selection algorithm will return false.
Transmitter Power (mW)	Local	1 to 100	It is the signal intensity of the transmitter. The higher the power radiated by the transmitter's antenna the greater the reliability of the communications system.
Antenna Gain (dBi)	Local	−1000 to 1000	A relative measure of an antenna's ability to direct or concentrate radio frequency energy in a particular direction or pattern. The measurement is typically measured in dBi (Decibels relative to an isotropic radiator).

Continued on next page

Table 2-4: *Data link layer, Physical layer and Network layer properties of Sensor. (continued).*

Parameter	Scope	Range	Description
Antenna Height (m)	Local	0 to 100	Antenna height is used in the pathloss calculation in the following models: Cost231 Hata Urban, Cost231 Hata SubUrban, Hata Urban, Hata SubUrban and Two Ray. This parameter has no effect when using any of the other pathloss models.
Power Source	Local	Main Line or Battery	Sensor communicate with each other using battery power. By default, the power model is set to Main Line, which represents a general-purpose alternating current (AC) electric power supply. The power model is user-configurable, with adjustable properties.
Energy Harvesting	Local	On or Off	Energy harvesting is the process of deriving energy from external sources (e.g., solar power, thermal energy, wind energy, and kinetic energy), capturing it, and storing it for use in small, wireless autonomous devices, such as those in wearable electronics and wireless sensor networks. NetSim supports an abstract energy harvesting model in which a specified amount of energy (calculated from the recharging current and specified voltage) is periodically added to the remaining energy of the node to replenish the battery. This feature can be turned on or off.
Initial Energy	Local	0.001–3250 mAh	A node has an initial value which is the level of energy the node has at the beginning of the simulation.
Transmitting Current	Local	0–10000 mA	In the Transmitting mode (Tx mode), the node consumes energy to transfer packets or data. The amount of energy consumed in this mode depends on the number of packets sent by the node, greater the number of packets, the more energy is consumed.
Idle Mode Current	Local	0–1000 mA	In idle mode, a node doesn't transmit or receive data but still listens to the wireless medium for potential packets and new nodes. This consumes less energy than sending or receiving, as no active communication occurs.
Voltage	Local	0–10 V	Voltage is a measure of the energy carried by the charge.
Receiving Current	Local	0–1000 mA	In the Receiving mode (Rx mode), the nodes are actively listening to the incoming data, it consumes the energy as it receives the data from the sender.
Recharging Current	Local	0–1000 mA	Recharging Current refers to the flow of electric charge supplied to a battery during the recharging process

Continued on next page

Table 2-4: *Data link layer, Physical layer and Network layer properties of Sensor. (continued).*

Parameter	Scope	Range	Description
SleepMode Current	Local	0–1000mA	Current flow during sleep mode.
Network Layer			
Routing Protocol	Global		RPL, AODV, DSR, ZRP, OLSR
RPL Routing Protocol			
Instance ID	Global	1 to 20	It is a unique identifier within a network. DODAGs with the same RPL Instance ID share the same Objective Function.
Node Type	Fixed	Router	A DAG root is a node within the DAG that has no outgoing edges. Because the graph is acyclic, all DAGs must have at least one DAG root and all paths terminate at a DAG root. LowPAN gateway is the root device. Any sensors can be configured as a Router or Leaf. If a node is configured to act as a router, it starts advertising the graph information with the new information to its neighboring peers. If the node is a "leaf node", it simply joins the graph and does not send any DIO message.
DAO Delay (s)	Global	0–20	A node should delay sending the DAO message to aggregate DAO information from other nodes for which it is a DAO parent. Receiving a DAO message starts the Delay DAO timer. DAO messages received while the Delay DAO timer is active do not reset the timer. When the Delay DAO timer expires, the node sends a DAO.
DIS Initial Delay (ms)	Global	1–1000	The time for which the node waits before sending DIS message.
DIS Interval (ms)	Global	1–1000	DIS Interval is the time interval between two DIS messages
DSR Routing Protocol			
ACK Type	Global	LINK_LAYER_ACK or NETWORK_LAYER_ACK	The user can enable either Link Layer ACK (Layer 2 ACK) or Network Layer ACK (Layer 3 ACK). Link Layer ACK uses MAC layer acknowledgment for route maintenance, while Network Layer ACK uses DSR acknowledgment for route maintenance. For more details, refer to sections 3.2.1 and 3.2.2 of the MANET Technology Library.
ZRP and OLSR Routing Protocol			
Hello Interval	Global	1–100 s	Hello interval parameter is used for neighbor discovery process. This parameter determines how frequently Hello messages are sent out and also how frequently a neighbor table will be updated.

Continued on next page

Table 2-4: *Data link layer, Physical layer and Network layer properties of Sensor. (continued).*

Parameter	Scope	Range	Description
Refresh Interval	Global	1–100 s	Refresh interval is the duration after which each active node periodically refreshes routes to itself.
IARP	Fixed		IARP is used by a node to communicate with the interior nodes of its zone and is limited by the zone radius.
TC Interval	Global	1–100 s	Topology Control messages are the link state signaling done by OLSR. These messages are sent at TC interval every time.
Zone radius	Global	2–225 m	Zone radius parameter is present for ZRP Protocol. ZRP divides the entire network into zones. The radius of these zones is defined by Zone radius.

2.3 Run Simulation

Click on the Run Simulation icon on the top toolbar.

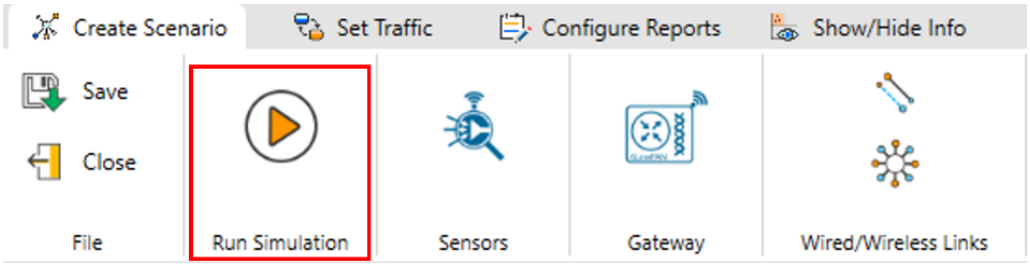


Figure 2-17: *Run Simulation option.*

Set the Simulation Time and click on Run.

3 Model Features

3.1 L3 Routing: DSR, OLSR, ZRP and AODV

Refer to the MANETs technology library (PDF) document for detailed information. Note that:

- WSN supports DSR, OLSR, ZRP and AODV protocols
- IOT supports AODV and RPL protocol, since only AODV and RPL have IPv6 support in NetSim.

3.2 L3 Routing: RPL Protocol

Routing Protocol for Low power and Lossy Networks (RPL) Overview

Low-power and Lossy Networks consist largely of constrained nodes (with limited processing power, memory, and sometimes energy when they are battery operated). These routers are interconnected by lossy links, typically supporting only low data rates that are usually unstable with relatively low packet delivery rates. Another characteristic of such networks is that the traffic patterns are not simply point-to-point, but in many cases point-to-multipoint or multipoint-to-point.

RPL Routing Protocol works in the Network Layer and uses IPv6 addressing. It runs a distance vector routing protocol based on Destination Oriented Directed Acyclic Graph (DODAGs).

Terminology of RPL routing protocol:

- DAG (Directed Acyclic Graph): A directed graph having the property that all edges are oriented in such a way that no cycles exist. All edges are contained in paths oriented toward and terminating at one or more root nodes.
- DAG root: A DAG root is a node within the DAG that has no outgoing edge. Because the graph is acyclic, by definition, all DAGs must have at least one DAG root and all paths terminate at a DAG root. In NetSim, only a single root is possible, i.e., the LowPAN Gateway.
- Destination-Oriented DAG (DODAG): A DAG rooted at a single destination, i.e., at a single DAG root (the DODAG root) with no outgoing edges.
- Up: Up refers to the direction from leaf nodes towards DODAG roots, following DODAG edges. This follows the common terminology used in graphs and depth-first search, where vertices further from the root are "deeper" or "down" and vertices closer to the root are "shallower" or "up".
- Down: Down refers to the direction from DODAG roots towards leaf nodes, in the reverse direction of DODAG edges. This follows the common terminology used in graphs and depth-first search, where vertices further from the root are "deeper" or "down" and vertices closer to the root are "shallower" or "up".
- Rank: A node's Rank defines the node's individual position relative to other nodes with respect to a DODAG root. Rank strictly increases in the Down direction and strictly decreases in the Up direction.
- RPLInstanceID: An RPL Instance ID is a unique identifier within a network. DODAGs with the same RPLInstanceID share the same Objective Function.
- RPL instance: When we have one or more DODAG, then each DODAG is an instance. An RPL Node may belong to multiple RPL Instances, and it may act as router in some and as a leaf in others. Any sensor can be configured as a Router or Leaf. Leaf nodes do not take part in RPL routing.
- DODAG ID: Each DODAG has an IPV6 ID. This ID is given to its root only. And as the root doesn't change the ID also doesn't change.
- Objective Function (OF): An OF defines how routing metrics, optimization objectives, and related functions are used to compute Rank. Furthermore, the OF dictates how parents in the DODAG are selected and, thus, the DODAG formation.

3.2.1 RPL Objective Function

The objective function in NetSim RPL seeks to find the route with the best link quality. The objective function:

`static UINT16 compute_candidate_rank(NETSIM_ID d, PRPL_NEIGHBOR neighbour)` can be found in Neighbor.c under the RPL project.

Link quality calculations, available in Zigbee Project 802.15.4.c file in function `get_link_quality()`.

$$L_q = \left(1 - \left(\frac{p}{rs}\right)\right)$$

where p = Received power (dBm) and rs = Receiver sensitivity (dBm)

And Final

$$Link\ Quality = \frac{Sending\ Link\ Quality + Receiving\ Link\ Quality}{2}$$

The rank calculations are done in Neighbour.c in the RPL project.

$$RankIncrease = (Max_{increment} - Min_{increment}) \times (1 - L_q)^2 + Min_{increment}$$

$$Rank = RankIncrease + Rank(Parent)$$

The link quality in this case is based on received power and can be modified by the user to factor in distance, delay, etc. Link quality is calculated by making calls to the functions in the following order:

1. `compute_candidate_rank()` – RPL\Neighbor.c
2. `fn_NetSim_stack_get_link_quality()` – NetSim network stack which in turn calls
3. `zigbee_get_link_quality()` – ZigBee\802_15_4.c

The function `fn_NetSim_stack_get_link_quality()` is part of NetSim's network stack which is closed to the user. However the function `zigbee_get_link_quality()` is open to the users and can be modified if required.

3.2.2 Topology Construction

NetSim IOT WSNs do not typically have predefined topologies, for example, those imposed by point-to-point wires, so RPL has to discover links and then select peers sparingly. RPL routes are optimized for traffic to or from one or more roots that act as sinks for the topology. As a result, RPL organizes a topology as a Directed Acyclic Graph (DAG) that is partitioned into one or more Destination Oriented DAGs (DODAGs), one DODAG per sink.

RPL identifiers: RPL uses four values to identify and maintain a topology:

- The first is an RPLInstanceID. An RPLInstanceID identifies a set of one or more Destination Oriented DAGs (DODAGs). A network may have multiple RPLInstanceIDs, each of which defines an independent set of DODAGs, which may be optimized for different Objective Functions (OFs) and/or applications. The set of DODAGs identified by an RPLInstanceID is called an RPL Instance. All DODAGs in the same RPL Instance use the same OF.
- The second is a DODAGID. The scope of a DODAGID is an RPL Instance. The combination of RPLInstanceID and DODAGID uniquely identifies a single DODAG in the network. An RPL Instance may have multiple DODAGs, each of which has a unique DODAGID.
- The third is a DODAGVersionNumber. The scope of a DODAGVersionNumber is a DODAG. A DODAG is sometimes reconstructed from the DODAG root, by incrementing the DODAGVersionNumber. The combination of RPLInstanceID, DODAGID, and DODAGVersionNumber uniquely identifies a DODAG Version.
- The fourth is Rank. The scope of Rank is a DODAG Version. Rank establishes a partial order over a DODAG Version, defining individual node positions with respect to the DODAG root.

DIO (DODAG Information Object) transmission:

DIO messages contain the information about the rank of the broadcasting node, the objective function(OF), DAG-ID, etc. Once a node receives a DIO, it calculates its rank based on the rank in the received DIO, and the link quality.

RPL nodes transmit DIOs using a Trickle Timer. This message is multicast downwards in a DODAG. With DIO, the child-parent relationship and sibling relationship are established.

DAO (Destination Advertisement Object):

A request for the root node (parent node) to join the network for communication which adds info of that node. The DAO is used to propagate destination information Upward along the DODAG.

DIS (DODAG Information Solicitation) transmission:

A new node wanting to join the DODAG checks whether a DODAG network exists. The nodes that are out of range exchange DIS messages to learn about the DODAG.

Rank:

- The Rank of a node is a scalar representation of the location of that node within a DODAG.
- The rank is a measure in some sense of the distance of the node from the root. It monotonically increases as we move away from the root.
- Rank is used to avoid and detect loops.
- The rank calculation is based on the objective function defined.
- The root node has a Rank of 1. This root node in IoT is also the border router.

DODAG Construction

- Nodes periodically send link-local multicast DIO messages.
- Stability or detection of routing inconsistencies influence the rate of DIO messages.
- Nodes listen for DIOs and use their information to join a new DODAG, or to maintain an existing DODAG.
- Exchange of DIO and DAO control messages is shown below.

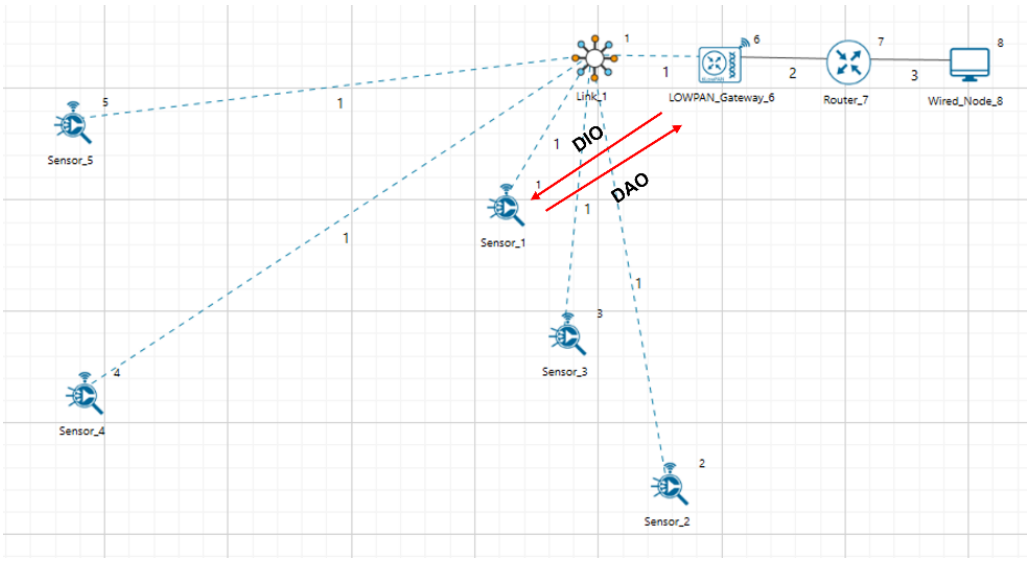


Figure 3-1: *DIO/DAO Control messages for Sensor and Lowpan gateway.*

- Rank is decided based on the DIO message received from the Root and the link quality.
- Based on information in the DIOs the node chooses parents to the DODAG root.
- As a result, the nodes follow the upward routes towards the DODAG root.
- If the destination is unreachable, then the root will drop the packet.
- Note that DIS messages are sent by sensors which are not part of the DODAG. The sensors which are part of the DODAG, and which received the DIO message will send the DAO message in return, whereas the sensors which did not receive the DIO messages will send the DIS message.

3.2.3 RPL Log File

Consider the above scenario as shown in Figure 3-1,

- Set pathloss as Log distance with a pathloss exponent of 3.5.
- This can be done by clicking on the link, which will open a right-hand panel where you can set the properties as discussed.
- Set Grid length as 100×100.
- Run the simulation for 10s.

Once the simulation is completed, users can access the rpl log file from the results dashboard as shown in Figure 3-2.

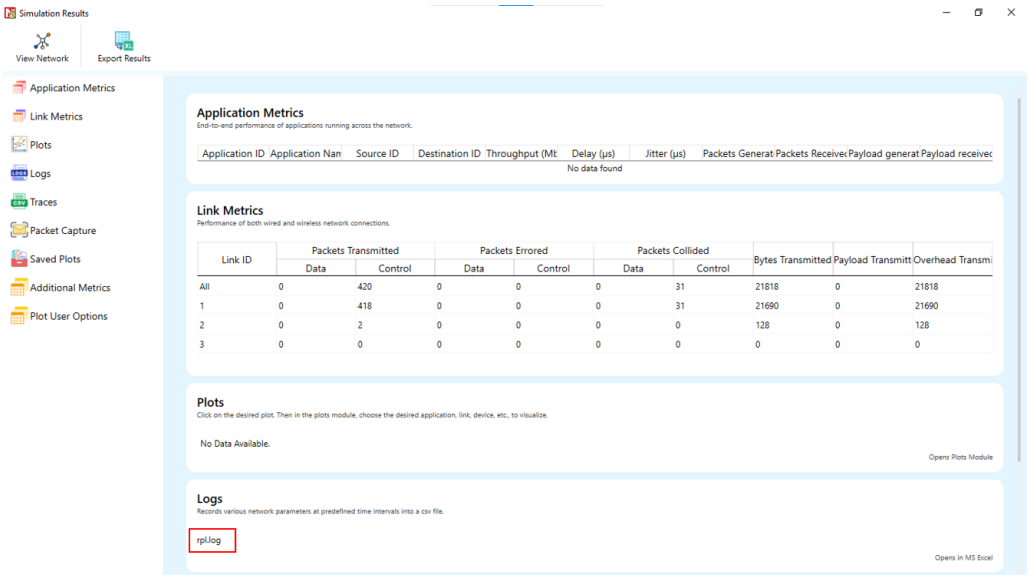


Figure 3-2: *Results dashboard window.*

However, to get detailed information related to Rank Calculations the DEBUG_RPL preprocessor directive needs to be uncommented in the code.

Procedure to get the RPL log file:

- Go to NetSim Home page and click on Your work.
- Click on Workspace Options and then click on Open Code and open the codes in Visual Studio. Set x86 or x64 according to the NetSim build which you are using.
- Go to the RPL Project in the Solution Explorer. Open RPL.h file and change `// #define DEBUG_RPL` to `#define DEBUG_RPL` as shown below:

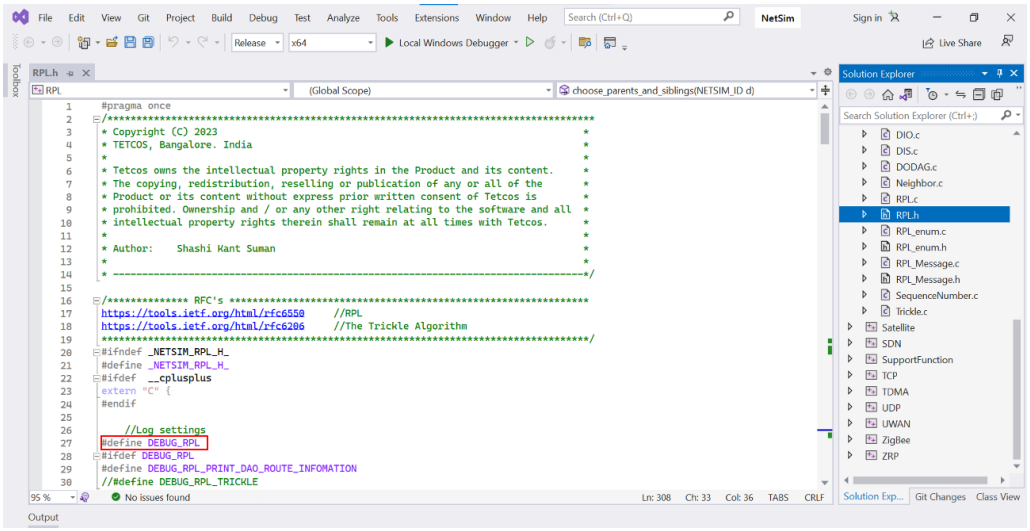


Figure 3-3: *Visual Studio.*

- Right click on the RPL project in the solution explorer and click on rebuild.
- After the RPL project is rebuilt successfully, go back to the network scenario.

Now after any IoT-RPL simulation, an RPL log file is generated with detailed information about the DODAG formation. An example rpl log file is shown below Figure 3-4.

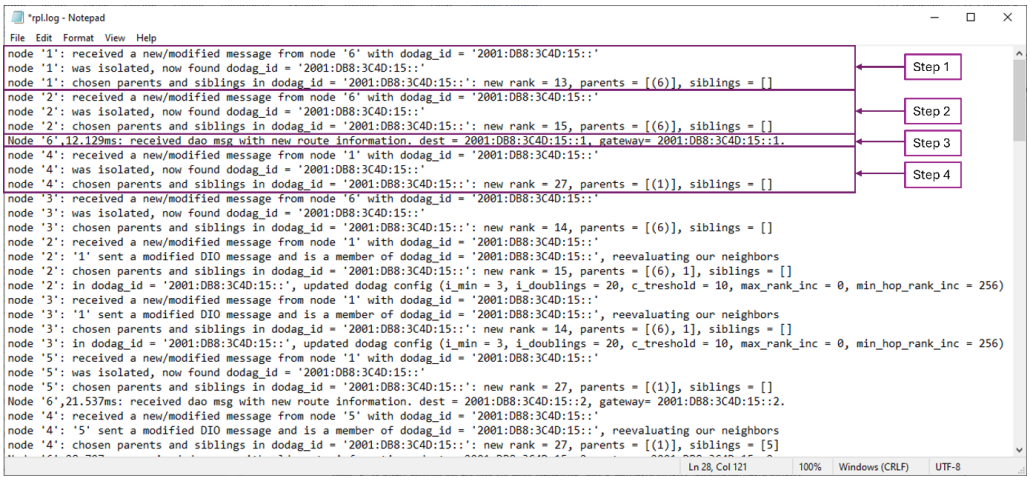


Figure 3-4: *RPL Log file.*

Explanation of the log file:

Step 1:

- Node 1 receives a DIO msg from Node 6 (i.e., root).
- Node 1 finds the DODAG id.
- Based on the DIO message received from Node 6, Node 1 chooses its “Parent as Node 6” and establishes its “New Rank = 13”. It does not have any siblings.

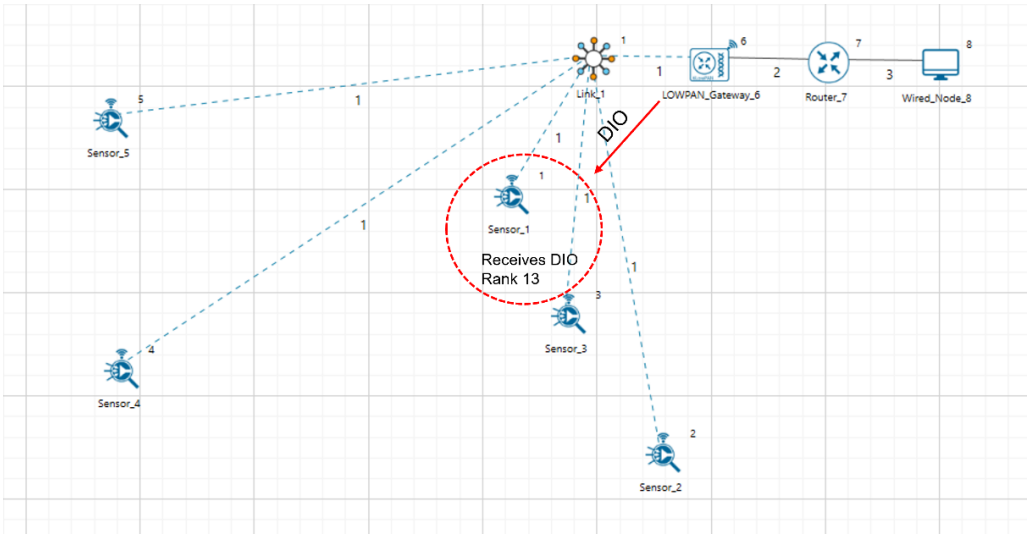


Figure 3-5: *Node 1 chooses its "Parent as Node 6" – New Rank = 13.*

Step 2:

- Node 2 receives a DIO msg from Node 6 (i.e. root).
- Node 2 finds the DODAG id.
- Based on the DIO message received from Node 6, Node 2 chooses its "Parent as Node 6" and establishes its "New Rank = 15". It does not have any siblings.

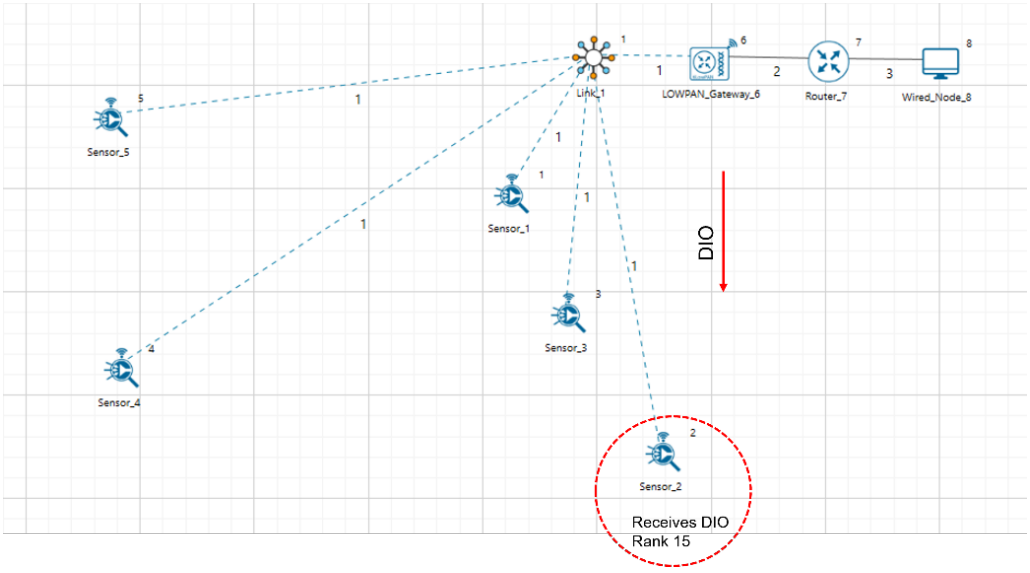


Figure 3-6: *Node 2 chooses its "Parent as Node 6" – New Rank = 15.*

Step 3:

- Node 6 receives a DAO message from Node 1 with the new route information about the destination and the Gateway.

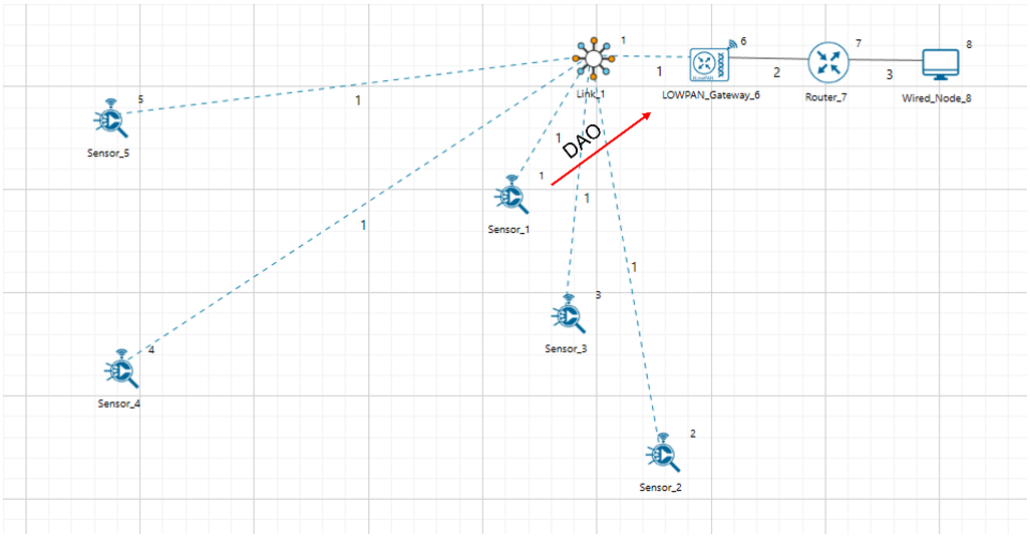


Figure 3-7: *Node 6 receives a DAO message from Node 1.*

Step 4:

- Node 4 receives a DIO msg from Node 1 (i.e. Sensor which is configured as Router).
- Node 4 finds the DODAG id.
- Based on the DIO message received from Node 1, Node 4 chooses its "Parent as Node 1" and establishes its "New Rank = 27" since it is in the next Rank level. It does not have any siblings.

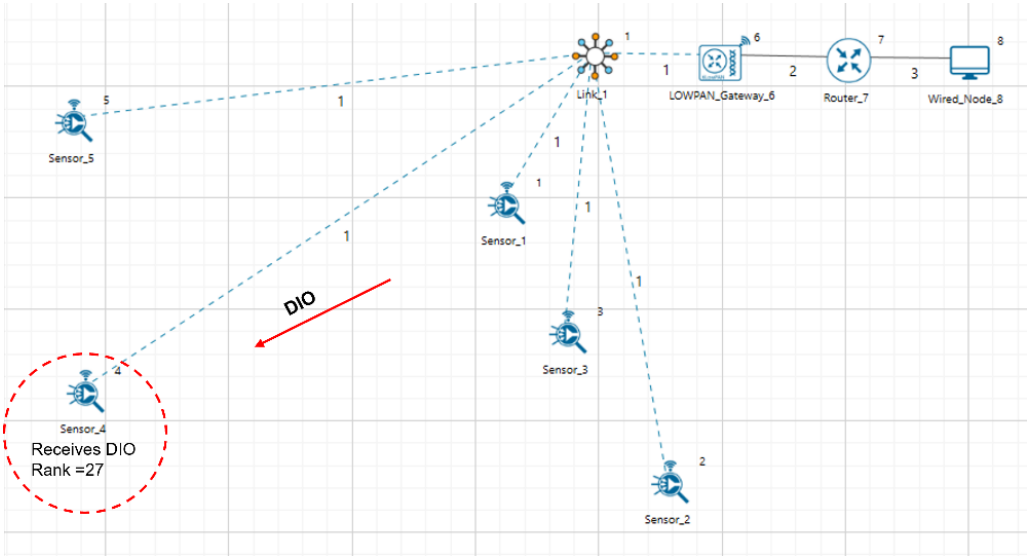


Figure 3-8: *Node 4 chooses its "Parent as Node 1" – New Rank = 27.*

Likewise, DODAG formation throughout the simulation is logged inside the rpl log file.

3.2.4 Viewing RPL control messages in Wireshark

Wireshark option can be enabled in the ZigBee devices to capture network traffic during the simulation. RPL control messages such as DAO, DIO etc. can be seen in Wireshark as shown in Figure 3-9.

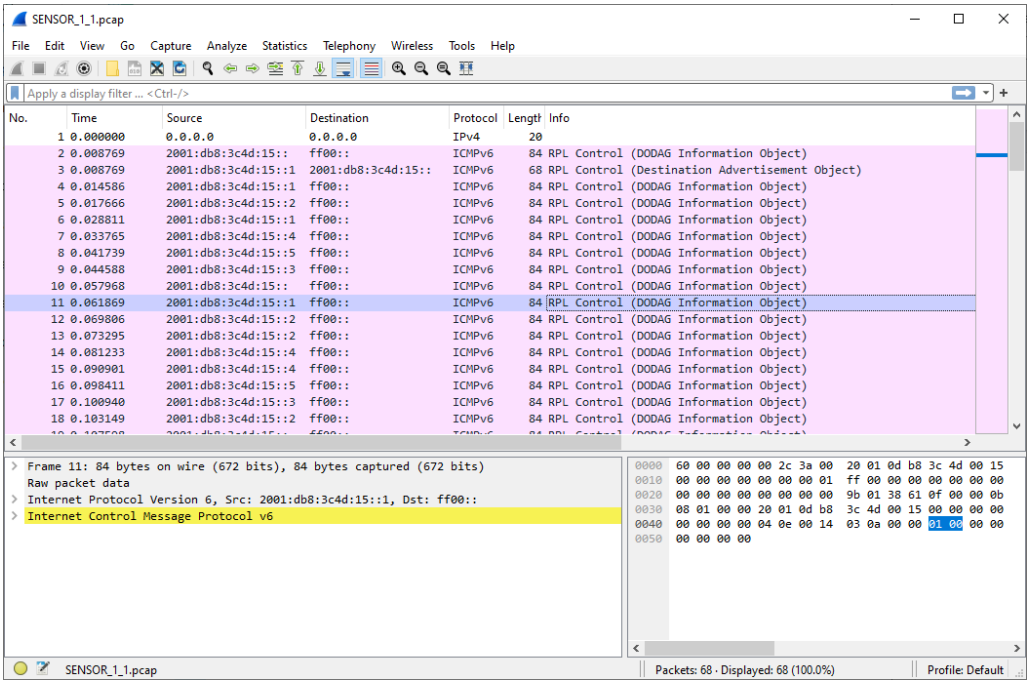


Figure 3-9: *Wireshark captures RPL control messages such as DAO, DIO etc.*

Following is a screenshot of a DIO message where the Rank information is highlighted as shown in Figure 3-10.

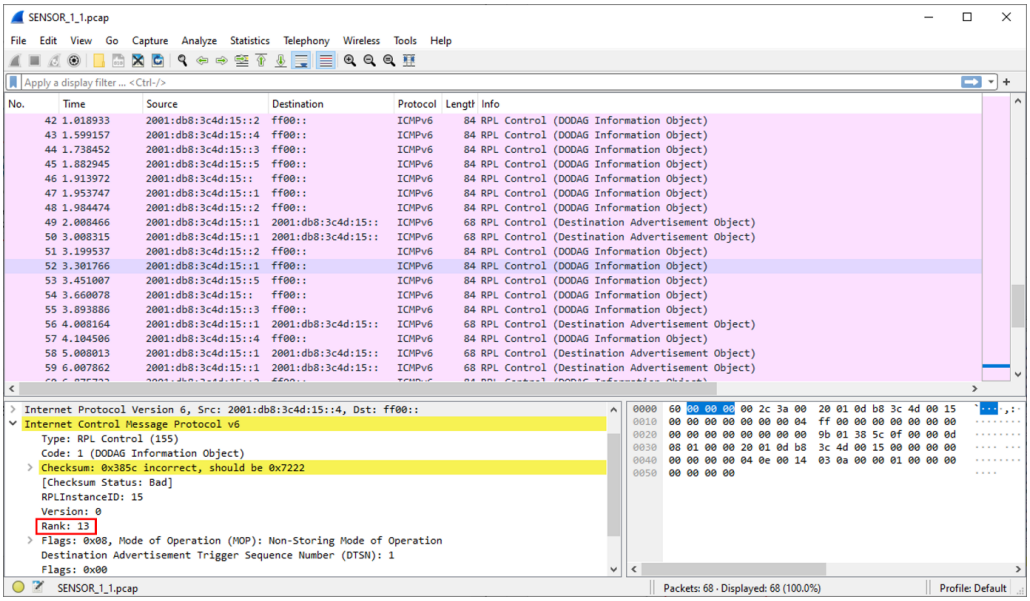


Figure 3-10: *DIO message with Rank information in Wireshark.*

3.3 MAC / PHY: 802.15.4 Overview

IEEE802.15/TG4 formulated the IEEE802.15.4 for low-rate wireless personal area network, i.e., LR-WPAN. The standard gives priority to low-power, low-rate and low-cost.

In NetSim, the WSN part of the IOT Network runs 802.15.4 in MAC and PHY. The features implemented are:

- PHY rate is fixed at 250 Kbps; it does not vary per the received SNR.

- Receive sensitivity is a GUI parameter modifiable by users; the default value is −85 dBm.
- Superframe
  - Beacon enabled and beacon disabled mode.
  - In beacon enabled mode NetSim supports slotted CSMA/CA with Active & Inactive Period (controlled by Beacon order and super-frame order parameters).
  - GTS is not implemented.
- Data Transfer Model
  - Device to coordinator, coordinator to device and device to device (peer to peer topology).
  - AckRequestFlag: If set the device acknowledges successful reception of the data frame by transmitting an ack frame.
- Frames
  - Beacon
  - Data
  - Acknowledgement
- CSMA / CA Mechanism
  - Non-beacon mode uses unslotted CSMA/CA.
  - Beacon mode uses slotted CSMA/CA.
- Energy Model
  - Energy sources: Main Line and Battery.
  - Energy Harvesting which uses recharging current to replenish battery energy.
  - Consumption Modes: Transmit, Receive, Idle and Sleep.

3.3.1 CSMA/CA Implementation in NetSim

- In both Slotted and Unslotted CSMA/CA cases, the CSMA/CA algorithm is based on backoff periods, where one backoff period is equal to aUnitBackoffPeriod which is 20 symbols long.
- This is the basic time unit of the MAC protocol and the access to the channel can only occur at the boundary of the backoff periods. In slotted CSMA/CA the backoff period boundaries must be aligned with the super-frame slot boundaries whereas in unslotted CSMA/CA the backoff periods of one device are completely independent of the backoff periods of any other device in a PAN.
- The CSMA/CA mechanism uses three variables to schedule the access to the medium:
  - NB is the number of times the CSMA/CA algorithm was required to backoff while attempting access to the current channel. This value is initialized to zero before each new transmission attempt.
  - CW is the contention windows length, which defines the number of backoff periods that need to be clear of channel activity before starting transmission. CW is only used with the slotted CSMA/CA. This value is initialized to 2 before each transmission attempt and reset to 2 each time the channel is assessed to be busy.
  - BE is the backoff exponent, which is related to how many backoff periods a device must wait before attempting to assess the channel activity.
- In beacon-enabled mode, each node employs two system parameters: Beacon order (BO) and Superframe Order (SO).
- The parameter BO decides the length of beacon interval (BI), where $BI = aBaseSuperframeDuration \times 2^{BO}$ symbols and $0 \leq BO \leq 14$; while the parameter SO decides the length of superframe duration (SD),

- where $SD = aBaseSuperframeDuration \times 2^{SO}$ symbols and $0 \leq SO \leq BO \leq 14$.
- The value of aBaseSuperframeDuration is fixed to 960 symbols. The format of the superframe is defined as shown in the following Figure 3-11.

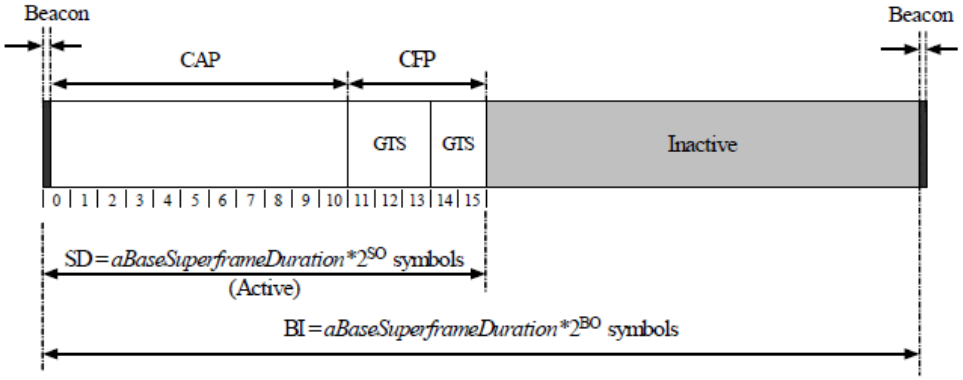


Figure 3-11: *The format of the Superframe structure.*

- Furthermore, the active portion of each Superframe consists of three parts: beacon, CAP, and CFP, which is divided into 16 equal length slots. The length of one slot is equal to $aBaseSlotDuration \times 2^{SO}$ symbols, where aBaseSlotDuration is equal to 60 symbols.
- In CAP, each node performs the CSMA/CA algorithm before transmitting data packet or control frame. Each node maintains three parameters: the number of backoffs (NB), contention window (CW), and backoff exponent (BE).
- The initial values of NB, CW, and BE are equal to 0, 2, and Min Backoff Expo, respectively, where Min Backoff Expo is by default 3 and it can be set up to 8.
- For every backoff period, node takes a delay for random backoff between 0 and $2^{BE} - 1$ Unit backoff Time (UBT), where UBT is equal to 20 symbols (or 80 bits).
- A node performs clear channel assessment (CCA) to make sure whether the channel is idle or busy, when the number of random backoff periods is decreased to 0.
- The value of CW will be decreased by one if the channel is idle; and the second CCA will be performed if the value of CW is not equal to 0. If the value of CW is equal to 0, it means that the channel is idle; then the node starts data transmission.
- However, if the CCA is busy, the value of CW will be reset to 2; the value of NB is increased by 1; and the value of BE is increased by 1 up to the maximum BE (Max Backoff Expo), where the value Max Backoff Expo is by default 5 and can be up to 8.
- The node will repeatedly take random delay if the value of NB is less than the value of Max CSMA BO (macMaxCSMABackoff), where the value of Max CSMA BO is equal to 4; and the transmission attempt fails if the value of NB is greater than the value of Max CSMA BO.

3.3.2 Beacon Order and Superframe Order

Beacon frame is one of the management frames in IEEE 802.15.4 based WSNs and contains all the information about the network. A coordinator in a PAN can optionally bound its channel time using a Superframe structure which is bound by beacon frames and can have an active portion and an inactive portion. The coordinator enters a low-power (sleep) mode during the inactive portion.

The structure of this Superframe is described by the values of macBeaconOrder and macSuperframeOrder. The MAC PIB attribute macBeaconOrder, describes the interval at which the coordinator shall transmit its beacon frames. The value of macBeaconOrder, BO, and the beacon interval, BI, are related as follows:

For $0 \leq BO \leq 14, BI = aBaseSuperframeDuration \times 2^{BO}$ symbols.

If BO = 15, the coordinator shall not transmit beacon frames except when requested to do so, such as on receipt of a beacon request command. The value of macSuperframeOrder, SO shall be ignored if BO = 15.

If SuperFrame Order (SO) is same as Beacon Order (BO) then there will be no inactive period and the entire SuperFrame can be used for packet transmissions. If BO=10, SO=9 half of the Superframe is inactive and so only half of Superframe duration is available for packet transmission. If BO=10, SO=8 then $\left(\frac{3}{4}\right)^{th}$ of the Superframe is inactive and so nodes have only $\left(\frac{1}{4}\right)^{th}$ of the Superframe time for transmitting packets.

3.4 Energy Models: Sources, Consumption and Harvesting

Wireless nodes, especially sensors, possess limited processing capability, storage and energy resources. The life of the sensor nodes i.e., the energy consumption during its operation, is critical to the network performance. Therefore, researchers often need to study energy consumption at the devices and in the network.

NetSim has dedicated energy models at sensor nodes (WSN/IoT networks) for modelling energy sources, energy consumption and energy harvesting.

The power model is user configurable and can be found in the ZigBee Interface properties of the Sensor nodes as shown in Figure 3-12. The default settings are as per Reference document [1].

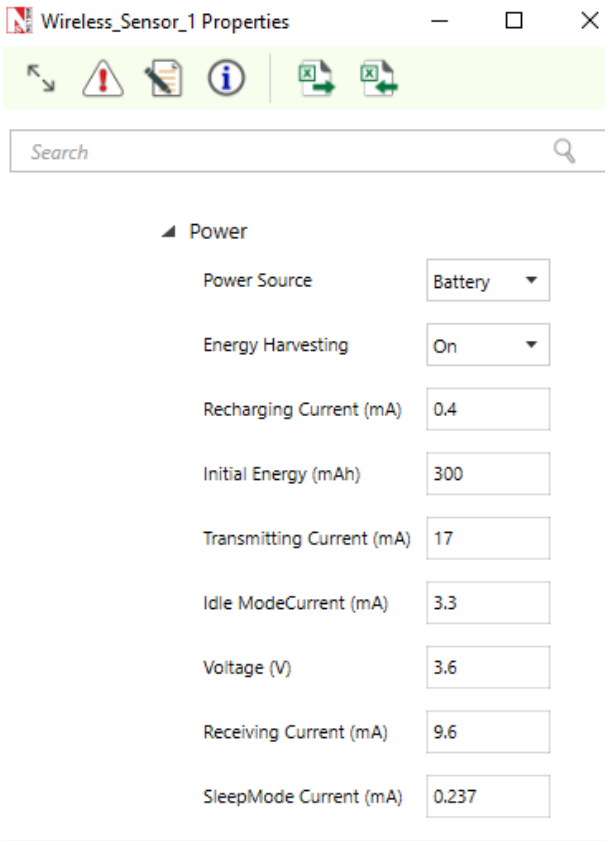


Figure 3-12: *Power model properties window.*

The power source represents the source of energy. Each node has its own single source of power. Main line power source is assumed to have infinite energy while batteries have limited initial energy. When energy harvesting is turned on, it replenishes the battery energy. If the power of a node is completely depleted the node can no longer operate.

NetSim assumes a fixed voltage and the different current drawn in the energy calculations are: Transmit

current, I_t, Receive current, I_R, Idle mode current, I_I, and Sleep mode current, I_S. This is a basic energy model and doesn't take into account battery non-linearities. The energy consumed in each of these activities would be

$$E_{TX} = I_t \cdot V \cdot T_{TX}$$

Where E_{TX} is the transmit energy, V is the battery voltage and T_{Tx} is the time spent for transmitting packets. And along similar lines

$$E_{RX} = I_R \cdot V \cdot T_{RX}$$

$$E_I = I_I \cdot V \cdot T_I$$

$$E_S = I_S \cdot V \cdot T_S$$

And the total energy consumed, E_{cons} is given by

$$E_{Cons} = E_{TX} + E_{RX} + E_I + E_S$$

NetSim also has an Energy-Harvesting Model given by

$$E_{EH} = I_{RC} \times V \times T_{RC}$$

where, E_{EH} is the harvested energy, I_{RC} is the recharging current and T_{RC} is the time for which the battery is recharged. Putting these together we get the expression

$$E^t = E_{init} - E^t_{Cons} + E^t_{EH}$$

Where E^t is the energy at time t, E^t_{Cons} is the energy consumed till time t, E^t_{EH} is the energy harvested till time t, and E_{init} is the initial energy of the battery.

Energy consumption is calculated individually for each sensor node that is part of the network scenario. The sensors have various Radio States such as SLEEP, TRX ON BUSY, RX ON IDLE, RX ON BUSY, RX OFF. As explained in the formulas above the energy consumed is proportional to the time for which the node is in a particular state. For example, the time for which a node transmits a packet is equal to the time for which the node is in TRX ON BUSY state. This duration in turn depends on the protocol operation. Thus, there is a correlation between protocol operation and energy consumption.

The units in NetSim for current is mA, for Voltage is V and for Total-Energy-Consumed is mJ. The Unit for Initial-Energy is mAh and this is converted to mJ for calculations since the output metrics are in mJ. The Initial energy in mAh is converted to mJ using the formula:

$$E_{init}\,[mJ] = E_{init}\,[mAh] \times V\,[V] \times 3600$$

For example, if we set $Initial\ energy = 0.5mAh$, and if the voltage in 1V then $E_{init}\,[mJ] = 1 \cdot 0.5 \cdot 3600 = 1800mJ$.

Post simulation NetSim outputs an Energy Metrics table which provides energy consumption of each device with respect to Transmission, Reception, Idle Mode, and Sleep Mode as shown in Figure 3-13.

Battery model

Device Name	Initial energy(mJ)	Consumed energy(mJ)	Remaining Energy(mJ)	Harvested Energy(mJ)	Transmitting energy(mJ)	Receiving energy(mJ)	Idle energy(mJ)	Sleep energy(mJ)
WIRELESS_SENSOR_1	1800.000000	340.279751	1499.719950	39.999700	12.168583	0.724992	327.386176	0.000000
WIRELESS_SENSOR_2	1800.000000	337.147674	1502.852027	39.999700	4.311657	6.789120	326.046897	0.000000
WIRELESS_SENSOR_3	1800.000000	335.426074	1504.573626	39.999700	7.753152	0.888115	326.784807	0.000000
WIRELESS_SENSOR_4	1800.000000	340.650365	1499.349335	39.999700	4.195039	12.237619	324.217707	0.000000
WIRELESS_SENSOR_5	1800.000000	333.209338	1506.790363	39.999700	4.606026	0.507494	328.095818	0.000000

Figure 3-13: *Battery model Table in result window showing various categories of energy consumption for each device. The categories are Initial energy, consumed energy, remaining energy, transmitting energy, receiving energy, idle energy and sleep energy.*

3.4.1 Energy Model source code

The Energy consumed by the sensor devices are computed in the function `battery_set_mode()` present in the BatteryModel.c file which belongs to the BatteryModel project. This is called in the function `fn_NetSim_Zigbee_ChangeRadioState()` present in the ChangeRadioState.c file which belongs to ZigBee project. The protocol operations decide the time for which the radio is in a particular state. The energy calculation function then multiplies this time by the current drawn and the voltage.

Users can implement energy aware protocols by accessing the information such as remaining energy of each node, when modifying the source code.

3.5 Sensor Application and how to model sensing interval?

Agents and sensing range were used in earlier versions (before v11) of NetSim as an abstraction of physical phenomenon to trigger packet generation in the sensor nodes respectively. Sensor nodes generate packets whenever they sense an agent within the sensor range. From NetSim v11 onwards users have the facility to configure traffic in the sensor network using the application as shown in Figure 3-14.

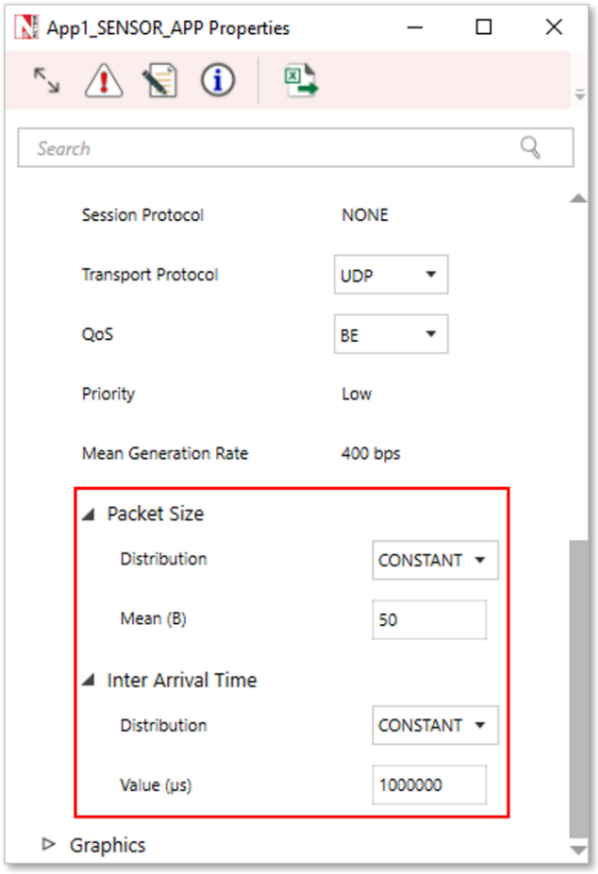


Figure 3-14: *Application window.*

In the application properties, the size of the packet can be set under packet size and the Inter-arrival time can be thought of as sensor interval.

Users can now configure traffic between sensor and sink node as well as between the sensor nodes.

Note that Agents, sensor interval, sensor range are deprecated in NetSim v11.

3.6 WSN/IOT File Based Placement

File based placement, as the name suggests is an option that can be used to place devices in user defined locations based on the text file which is provided as the input.

Why do we need File Based Placement?

- File Based Placement gives a completely user-defined approach for device placements during the process of Network design.
- This feature allows the user to design a large network scenario comprising of various devices with ease.
- It allows device placements with precision and so on.

Create a text file as per the following or use the file present in the Docs folder of NetSim Install Directory `< C:\Program Files\NetSim Standard\Docs\Sample Configuration\IOT>`

3.6.1 Internet of Things

The text file that we give as an input can be saved as follows: IOT File Based Placement.txt

The general format to be followed while creating an IOT File Based Placement.txt for all the devices used in it is given below:

```
<DEVICE NAME>,<DEVICE TYPE>,<X>,<Y>
```

where,

DEVICE NAME represents the name of the device and can be user defined.

DEVICE TYPE represents the type of device, and this info can be obtained from the "General Properties" of that particular device.

X represents the X Coordinate position of the device upon the grid.

Y represents the Y Coordinate position of the device upon the grid.

NOTE: Once we give a file-based input for device placement, an ad-hoc link will automatically be established connecting all the devices pertaining to it. And users need to manually connect the remaining devices using the Wired/Wireless links.

Must the IOT text file contain only IOT devices?

The IOT csv file can include all the devices that are present in the top ribbon/toolbar when we select Internet of Things from the home screen. This varies based on the network type. For e.g. WSN and MANETs network types support different devices comparatively.

IOT File based placement.csv

```
Format: Device_Name,Device_Type,X,Y
Wireless Sensor,Sensors,0,0
Wireless Sensor,Sensors,10,10
Wireless Sensor,Sensors,20,20
Wireless Sensor,Sensors,30,30
Wireless Sensor,Sensors,40,20
Low Pan Gateway,GateWay,50,10
```


Open NetSim and click New Simulation → Internet Of Things. In the Fast Config window, Choose the File Based Placement option under Automatic Placement and give the path of the text file as shown in Figure 3-15.

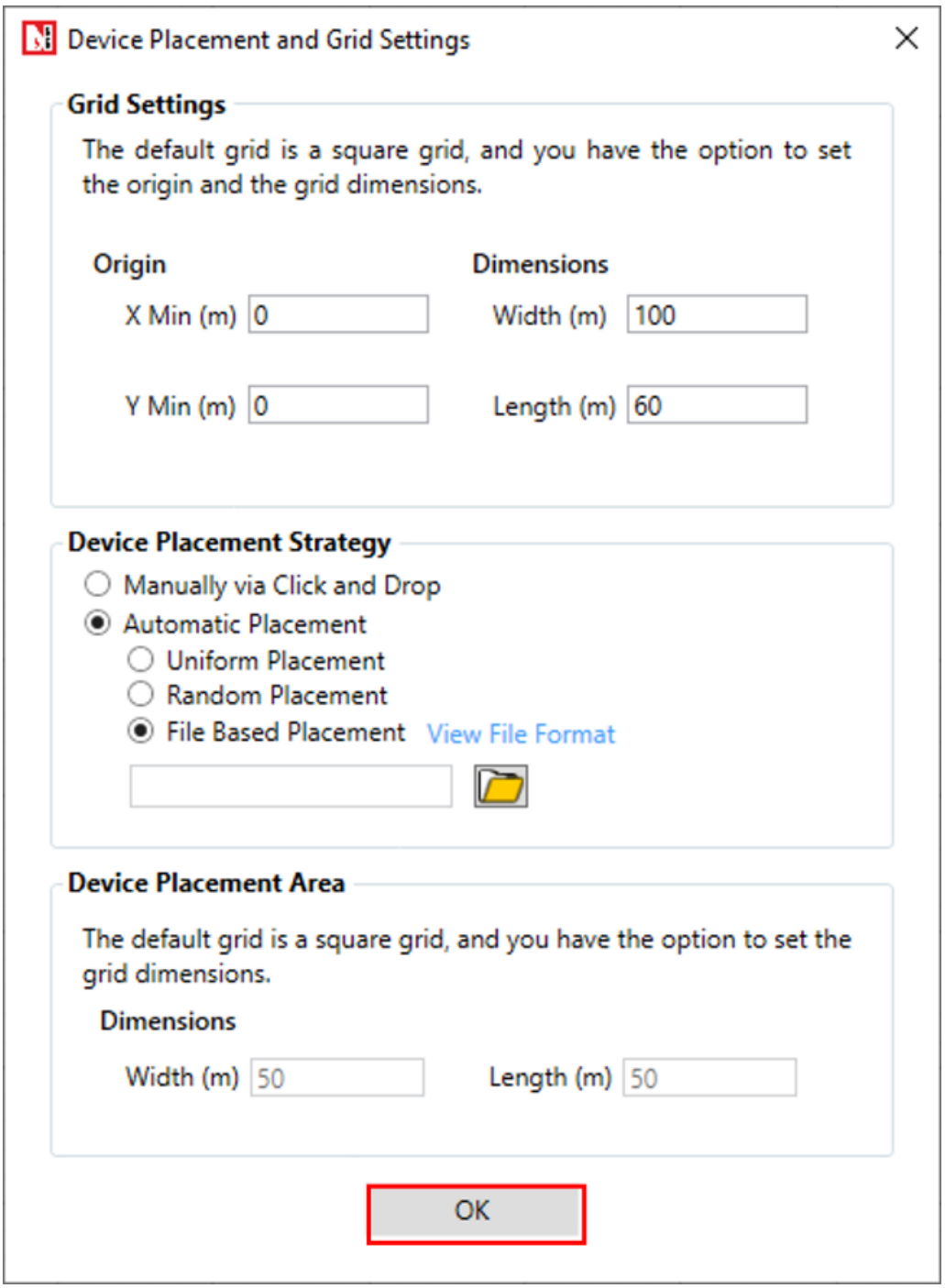


Figure 3-15: *Device placement Strategy to File based Placement in IOT.*

After giving the path, Click on OK. It will display the IOT network as shown below, where all devices are placed as per the positions given in the text file.

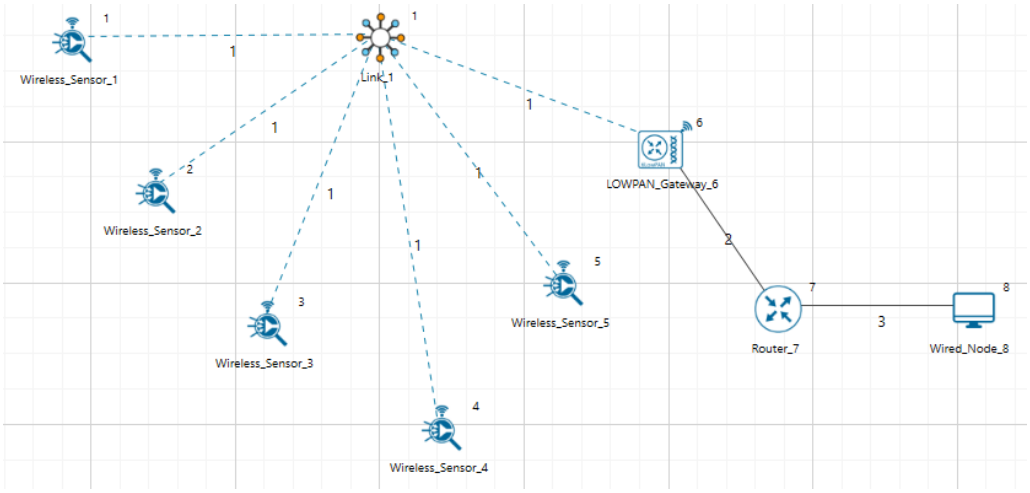


Figure 3-16: *Network Topology in IOT.*

Connect Low Pan Gateway to Router and Router to Wired node. Configure application and run simulation.

3.6.2 Wireless Sensor Networks

Create a csv file as per the following or use the file present in the Docs folder of NetSim Install Directory `< C:\Program Files\NetSim Standard\Docs\Sample_Configuration\WSN>`

```
WSN_File_based_placement.csv
Format: Device_Name,Device_Type,X,Y
Wireless_Sensor,Sensors,5,5
Wireless_Sensor,Sensors,10,10
Wireless_Sensor,Sensors,20,10
Wireless_Sensor,Sensors,5,10
WSN_Sink,sinknode,10,15
```

Open NetSim and click New Simulation → Wireless Sensor Networks. Select File Based Placement option under Automatic Placement and give the path of the text file as shown in Figure 3-17.

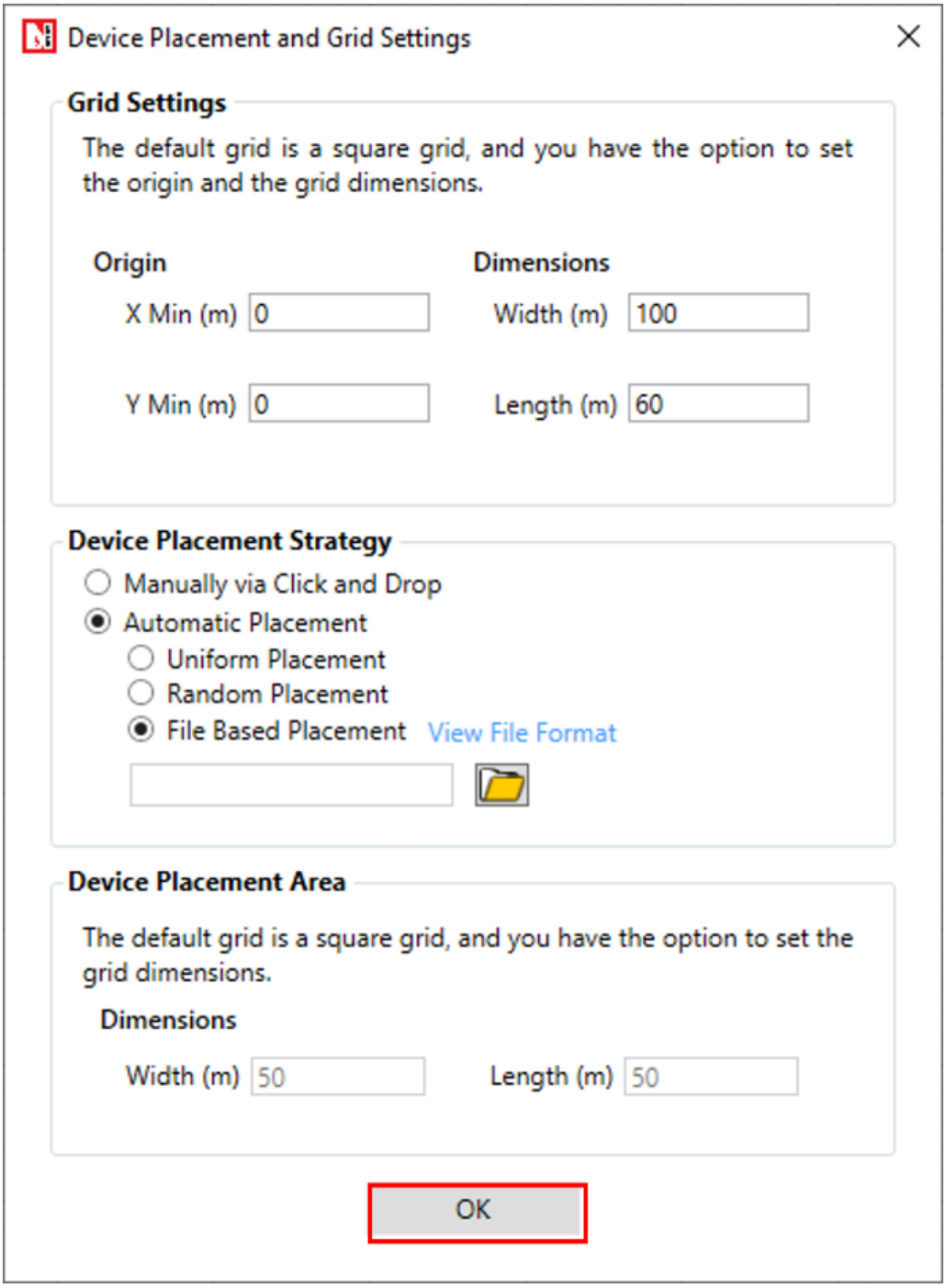


Figure 3-17: *Device placement Strategy to File based Placement in WSN.*

After giving the path, Click on OK. It will display the WSN network as shown below, where all devices are placed as per the positions given in the text file.

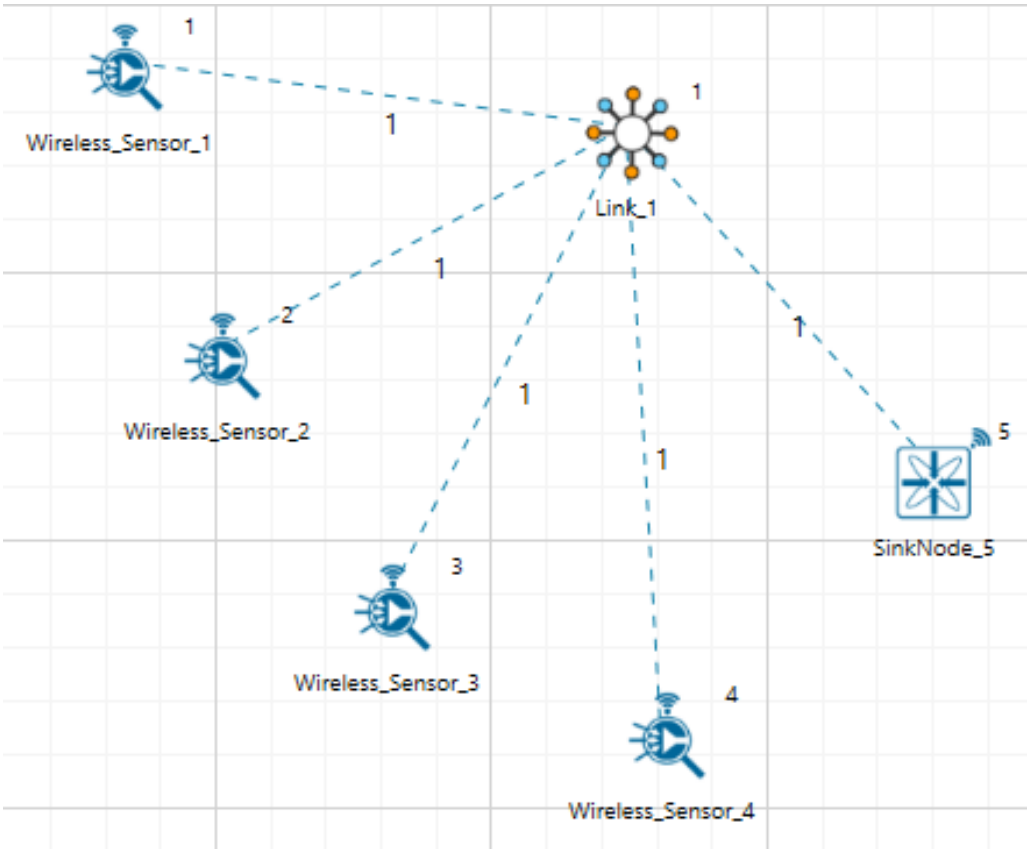


Figure 3-18: *Network Topology in WSN.*

NOTE: Please refer to section "2.1 Fast configuration" for more information.

3.7 Radio measurements log file

NetSim IEEE 802.15.4 Radio Measurements csv log file records pathloss, shadowing loss, fading loss, received power, transmitted power, SNR, BER. This log can be enabled by clicking on configure reports in top ribbon ▷ Plots ▷ Select IEEE 802.15.4 Radio Measurements under Network Logs as shown below.

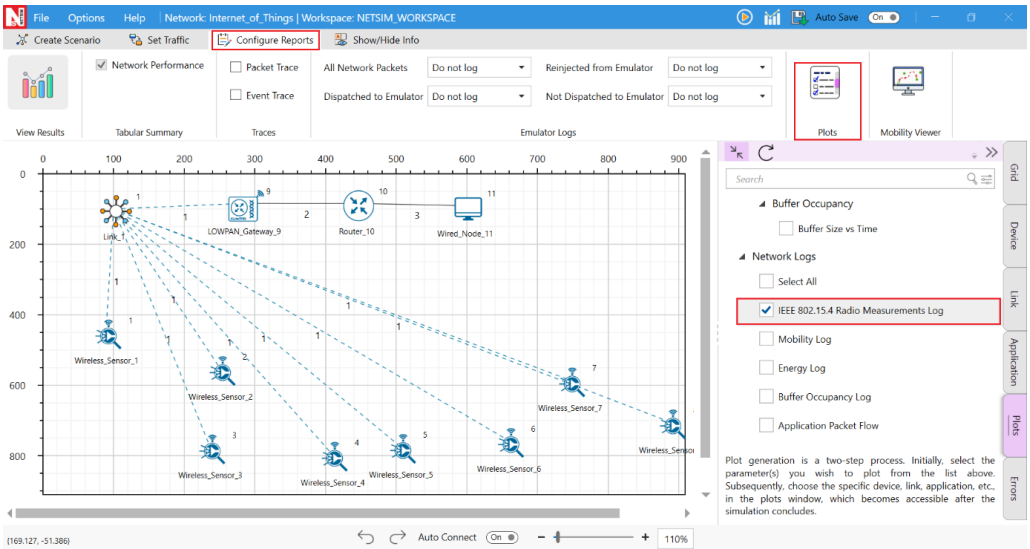


Figure 3-19: *Enabling IEEE 802.15.4 Radio Measurements.*

The IEEE802.15.4 Radio Measurements log.csv file will contain the following information:

- Time in Millisecondss
- Transmitter Name
- Receiver Name
- Distance between the Transmitter and the Receiver in meters
- Packet ID
- Packet Type
- Control Packet Type
- Transmitter Power in dBm
- Total Loss in dB
- Pathloss in dB
- Shadowing Loss in dB
- Fading Loss in dB
- Received Power in dBm
- SNR in dB
- BER

The log file can be accessed from the Simulations Results Window under the logs as shown below.

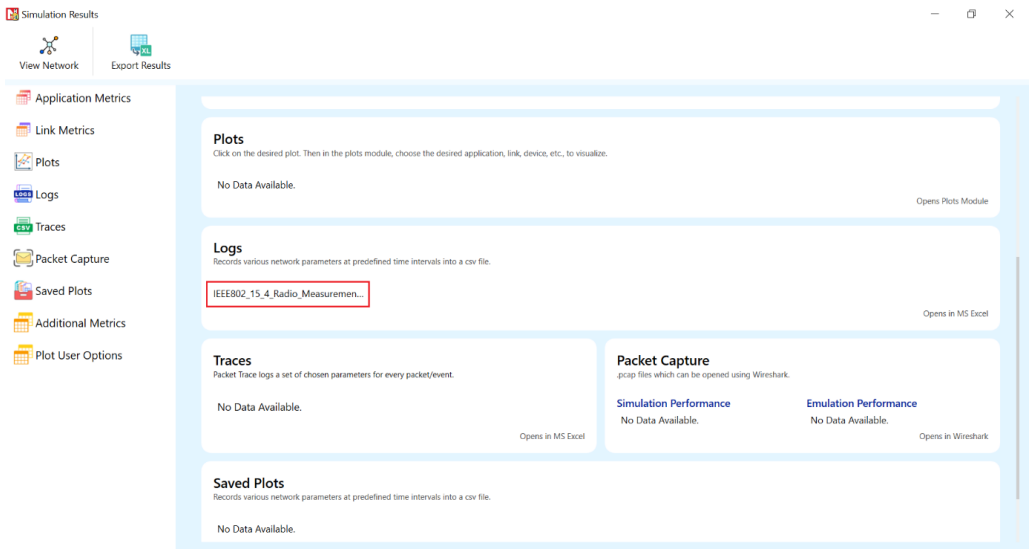


Figure 3-20: *IEEE802.15.4 RADIO MEASUREMENTS LOG.csv file highlighted in the Results window.*

Time(µs)	Transmitter Name	Receiver Name	Distance(m)	Packet ID	Packet Type	Control Packet Type	Tx_Power(dBm)	Path Loss(dB)	Shadowing Loss(dB)	Fading Loss(dB)
512	WSN_SINK_3	WIRELESS_SENSOR	150.163828	0	Control_Packet	Zigbee_BEACON_FRAM	0	83.5776	0	0
512	WSN_SINK_3	WIRELESS_SENSOR	89.366432	0	Control_Packet	Zigbee_BEACON_FRAM	0	79.069782	0	0
985504	WIRELESS_SENSOR_1	WIRELESS_SENSOR	145.92	0	Control_Packet	DSR_RREQ	0	83.32859	0	0
985504	WIRELESS_SENSOR_1	WSN_SINK_3	150.163828	0	Control_Packet	DSR_RREQ	0	83.5776	0	0
989344	WIRELESS_SENSOR_1	WIRELESS_SENSOR	145.92	0	Control_Packet	DSR_RREQ	0	83.32859	0	0
992192	WSN_SINK_3	WIRELESS_SENSOR	150.163828	0	Control_Packet	DSR_RREP	0	83.5776	0	0
992576	WIRELESS_SENSOR_1	WSN_SINK_3	150.163828	0	Control_Packet	Zigbee_ACK	0	83.5776	0	0
995552	WIRELESS_SENSOR_2	WIRELESS_SENSOR	145.92	0	Control_Packet	DSR_RREQ	0	83.32859	0	0
995552	WIRELESS_SENSOR_2	WSN_SINK_3	89.366432	0	Control_Packet	DSR_RREQ	0	79.069782	0	0
1000992	WIRELESS_SENSOR_1	WSN_SINK_3	150.08331	1	Sensing	App1_SENSOR_APP	0	83.572942	0	0
1001376	WSN_SINK_3	WIRELESS_SENSOR	150.08331	0	Control_Packet	Zigbee_ACK	0	83.572942	0	0
1005152	WIRELESS_SENSOR_1	WSN_SINK_3	150.08331	2	Sensing	App1_SENSOR_APP	0	83.572942	0	0
1005536	WSN_SINK_3	WIRELESS_SENSOR	150.08331	0	Control_Packet	Zigbee_ACK	0	83.572942	0	0
1008640	WSN_SINK_3	WIRELESS_SENSOR	89.022469	0	Control_Packet	DSR_RREP	0	79.036287	0	0
1009024	WIRELESS_SENSOR_2	WSN_SINK_3	89.022469	0	Control_Packet	Zigbee_ACK	0	79.036287	0	0
1012800	WIRELESS_SENSOR_2	WIRELESS_SENSOR	146	0	Control_Packet	DSR_RREP	0	83.333351	0	0
1013184	WIRELESS_SENSOR_1	WIRELESS_SENSOR	146	0	Control_Packet	Zigbee_ACK	0	83.333351	0	0
2003872	WIRELESS_SENSOR_1	WSN_SINK_3	150.08331	3	Sensing	App1_SENSOR_APP	0	83.572942	0	0
2004256	WSN_SINK_3	WIRELESS_SENSOR	150.08331	0	Control_Packet	Zigbee_ACK	0	83.572942	0	0
3003872	WIRELESS_SENSOR_1	WSN_SINK_3	150.08331	4	Sensing	App1_SENSOR_APP	0	83.572942	0	0

IEEE802_15_4_RADIO_MEASUREMENTS | Pivot Table(Custom)

Figure 3-21: *IEEE802.15.4 Radio Measurements log.csv file.*

3.7.1 Implementation details and Assumptions

The log is written during each packet received at the physical layer (PHY IN). Hence the number of entries will be based on the number of packets received, by all nodes or specific nodes based on values present in the DEVICE ID LIST array.

3.8 Energy log file

NetSim Energy log file records Current Time, Mode Switch Time, Mode, Device Name, Device Id, Voltage, Mode Current, Initial Energy, Remaining Energy, Transmit Energy, Receive Energy, Idle Energy, Sleep Energy, Harvested Energy, Consumed Energy.

The Energy log can be enabled by clicking on the icon present in the tool bar as shown below.

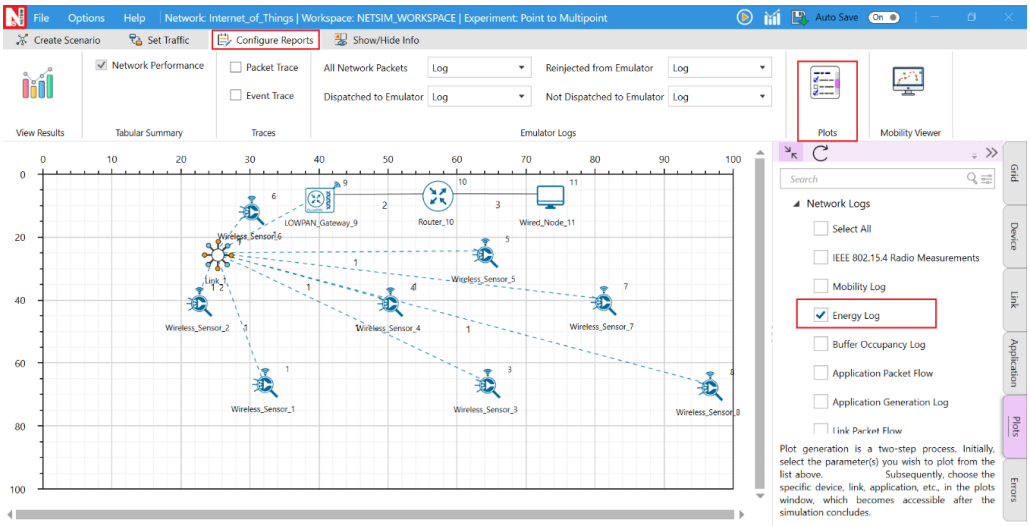


Figure 3-22: *Enabling Energy Log.*

The Energy Log.csv file will contain the following information:

- Current Time in Milliseconds
- Mode Switch Time in Milliseconds
- Mode, Current device Mode
- Device Name

- Device Id
- Voltage
- Mode Current, Current associated with Mode
- Initial Energy in Millijoules
- Remaining Energy in Millijoules
- Transmit Energy in Millijoules
- Receive Energy in Millijoules
- Idle Energy in Millijoules
- Sleep Energy in Millijoules
- Harvested Energy in Millijoules
- Consumed Energy in Millijoules

The log file can be accessed from the Simulations Results Window under the log file drop down in the left pane.

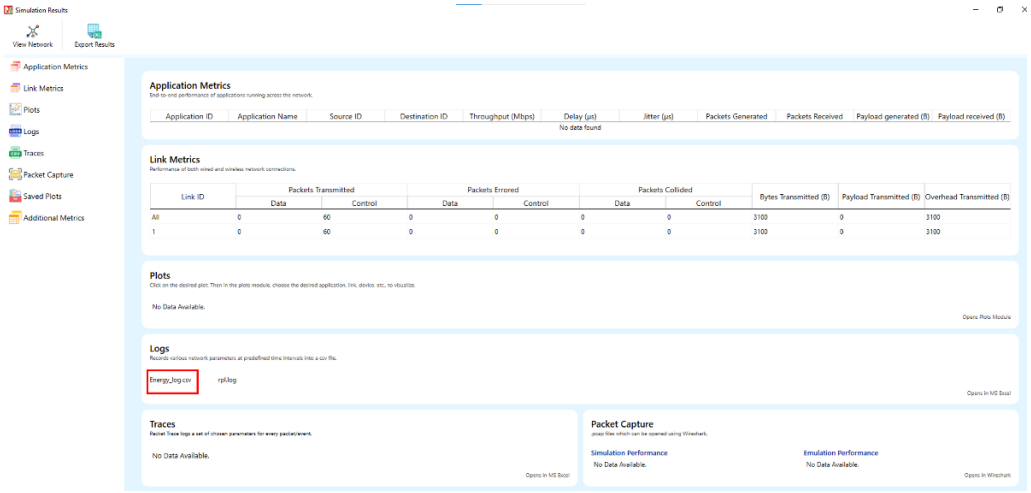


Figure 3-23: *Energy Log file highlighted in the Results window.*

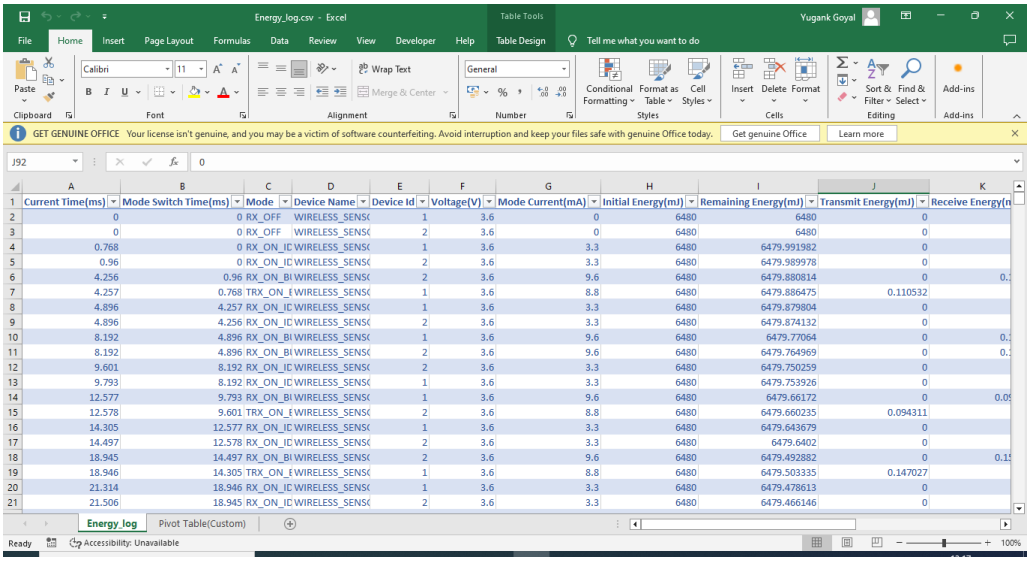


Figure 3-24: *Energy Log.csv file.*

Implementation details and Assumptions:

- Mode Switch Time indicates the time when the device is set to current Mode.
- Mode Current, Current associated with Mode.

3.9 Mobility Log File

NetSim mobility log file records Time, Device Name, Device Id, Position X, Position Y, Position Z. The Mobility log can be enabled by clicking on icon present in the tool bar as shown below.

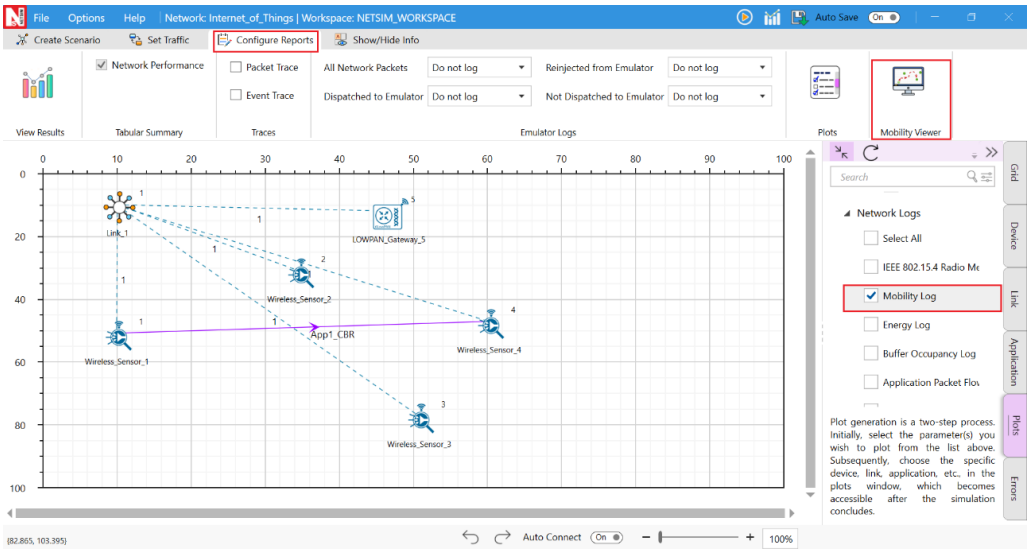


Figure 3-25: *Enabling Mobility Log.*

The Mobility Log.csv file will contain the following information:

- Time in Milliseconds
- Device Name
- Device Id
- Position X(m)
- Position Y(m)
- Position Z(m)

The log file can be accessed from the Simulations Results Window under the log file drop down in the left pane.

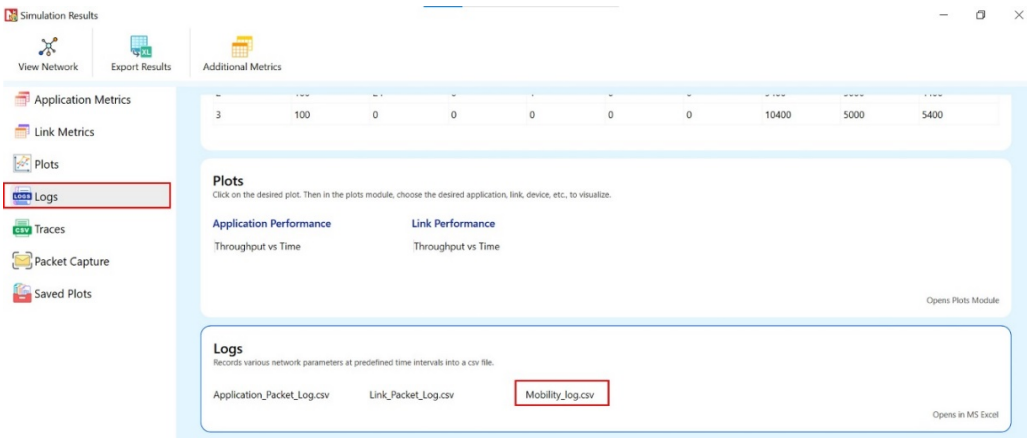


Figure 3-26: *Mobility Log file highlighted in the Results window.*

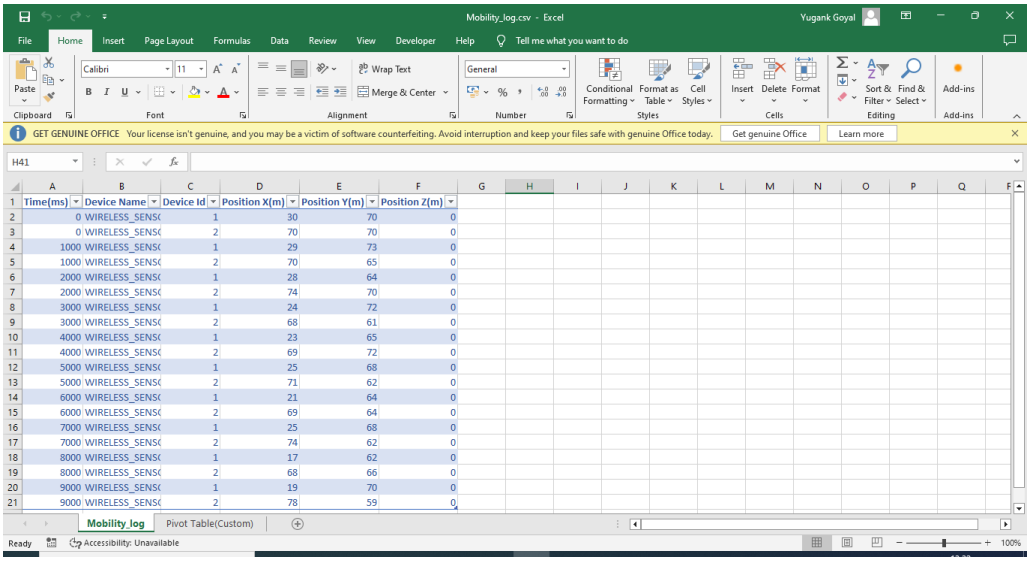


Figure 3-27: *Mobility Log.csv file.*

Additionally, the mobility path for sensors can be observed in the design window using mobility viewer option as shown in the screenshot below. For more details on the mobility viewer refer the user manual section 8.6.

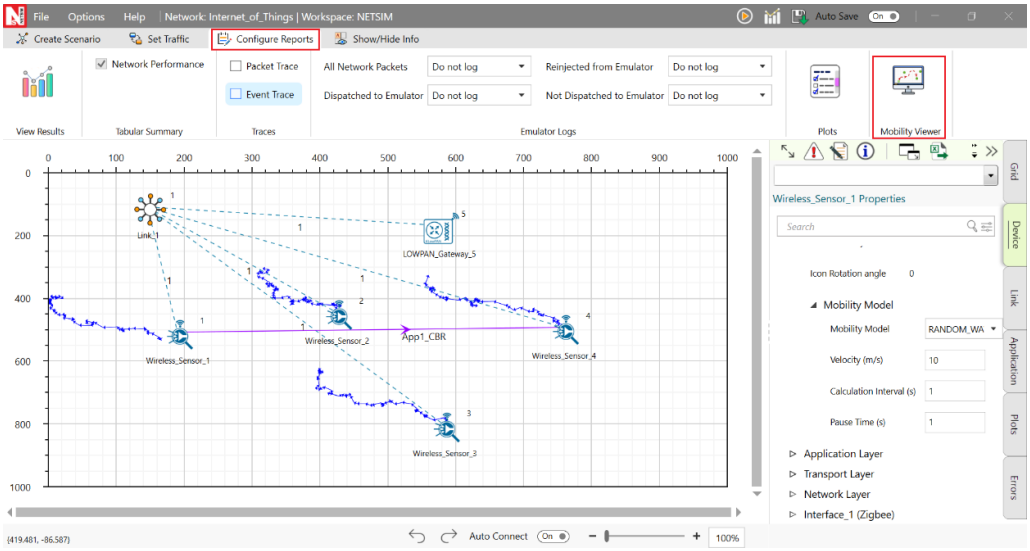


Figure 3-28: *Mobility path showing for sensors.*

3.10 Model Limitations

- NetSim currently supports only a single root in RPL.
- NetSim GUI supports only one RPL instance. Multiple RPL instances can be created by manually editing the config file.
- DODAG Repair is not supported.
- Nodes (sensors) in NetSim retrieve time from the same (single) virtual clock that ticks virtual time within NetSim. As such, they can be considered as perfectly time synchronized. Real world clocks drift from a reference clock due to various reasons (heat, deficient oscillator, etc.,), resulting in an offset of a few milliseconds per day. In the COTS version of NetSim, clock drift is not available. This means that it is not possible to model clocks to run with different rates and/or offsets, in different nodes/sensors.
- Security in 802.15.4 is not implemented.

4 Featured Examples

Sample configuration files for all networks are available in the Examples Menu in NetSim Home Screen. These files provide examples on how NetSim can be used – the parameters that can be changed and the typical effect it has on performance.

4.1 IOT

4.1.1 Energy Model

Open NetSim, Select Examples ▷IOT-WSN ▷Internet of Things ▷Energy Model then click on the tile in the middle panel to load the example as shown in Figure 4-1.

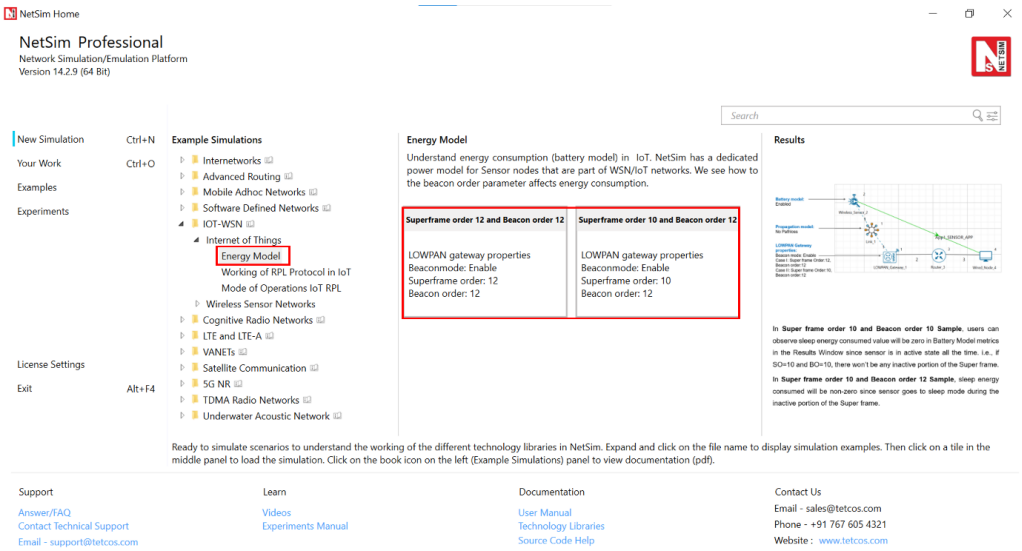


Figure 4-1: *List of scenarios for the example of Energy Model.*

The following network diagram illustrates what the NetSim UI displays when you open the example configuration file.

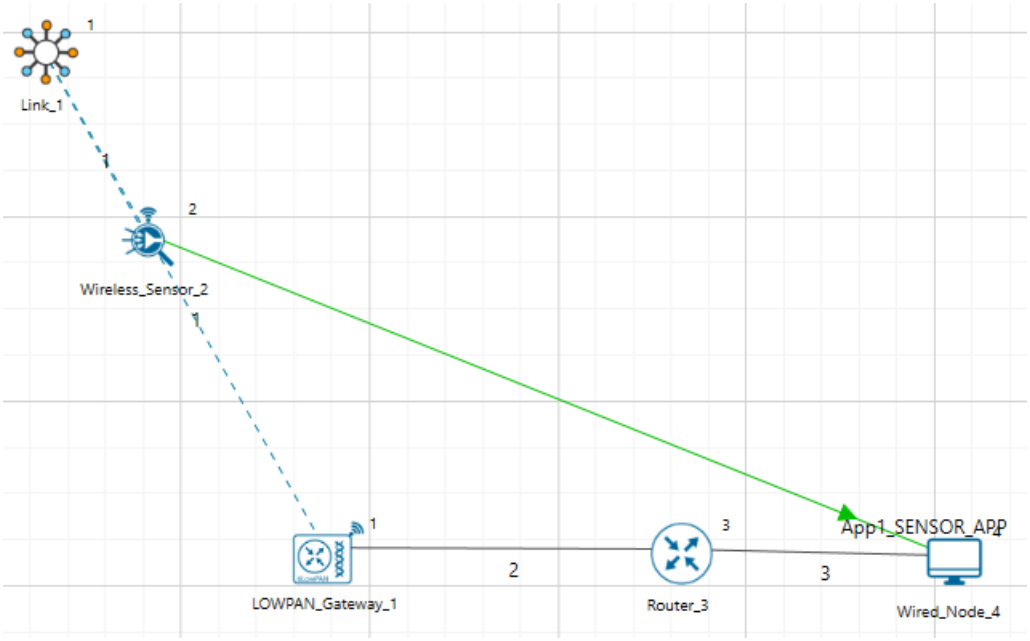


Figure 4-2: *Network set up for studying the Energy Model.*

Settings done in sample network:

1. Grid is set to 1000 × 500 from grid setting property panel on the right. This needs to be done before any device is placed on the grid.
2. Click on LOWPAN Gateway and expand the right property panel, enable the Beacon Mode and change the Superframe Order to 12 and the Beacon Order to 12 under the Interface 1 (Zigbee) layer.

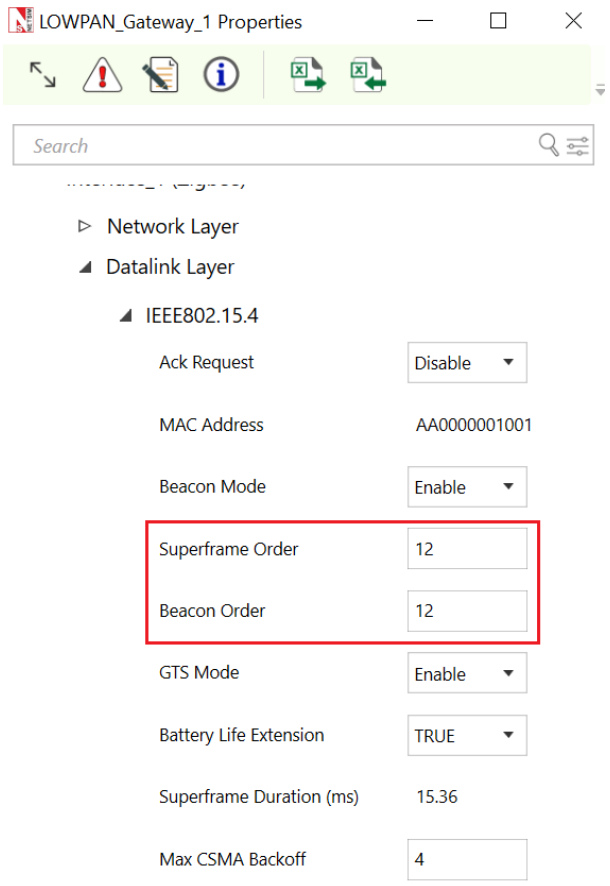


Figure 4-3: *Datalink layer Properties for Lowpan Gateway.*

3. Click on sensor, go to physical layer in Interface (Zigbee) layer and change the power source to Battery.
4. Click on link and open property panel on the right and set the channel characteristics → No pathloss in Ad hoc link Properties.
5. Configure a sensor application between Sensor 2 and Wired node 4 from the Set Traffic tab in ribbon on the top.
6. Enable the sleep energy vs time plot by clicking on configure reports tab ▷ Plots and run the simulate for 50 sec.
7. In Simulation metrics window, click on Additional metrics in the left panel and check the Battery model metrics, users should get zero value for Sleep energy (mJ) consumed.
8. Go back to Simulation window change following properties in LOWPAN Gateway for another sample.
9. Set Superframe Order (SO) and Beacon Order (BO) as 10 and 12 respectively ($0 \leq SO \leq BO \leq 14$).
10. Re-run the Simulate for 50 sec.

Results:

1. In the simulation results window click on additional metrics on the left and scroll down for the Battery model metrics for both the cases.

In Superframe order 12 and Beacon order 12 Sample, users can observe sleep energy consumed value will be zero in Battery model metrics since sensor is in active state all the time, i.e., if SO=12 and BO=12, there won't be any inactive portion of the Superframe.

Battery model

Device Name	Initial energy(mJ)	Consumed energ	Remaining Energ	Harvested Energy	Transmitting ener	Receiving energy	Idle energy(mJ)	Sleep energy(mJ)
WIRELESS_SENSOR_	6480.000000	587.825417	5962.747624	70.573041	6.985947	1.876746	578.962724	0.000000

Figure 4-4: *Battery Model Table for SO=12 & BO=12.*

Similarly, open the Sleep energy vs time plot from simulation results window, compare the sleep energy consumed by sensor over the simulation time for both cases when SO =12 and BO =12 and SO =10 and BO =12.

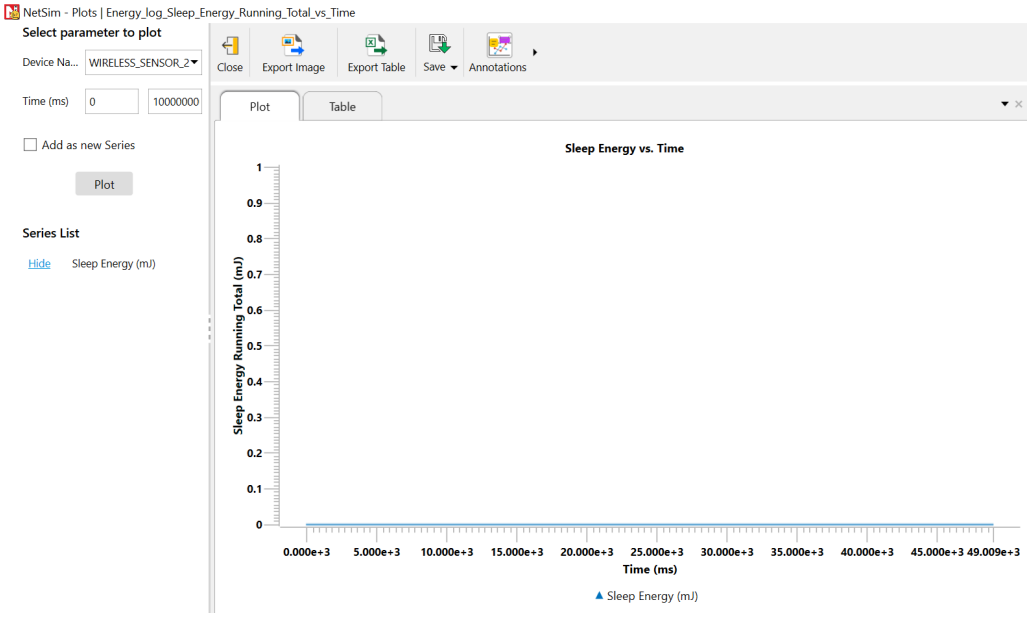


Figure 4-5: *The above plot shows the sleep energy consumed when SO=12 and BO=12. The sleep energy consumed is zero since there is no inactive portion in the beacon interval, and the sensors remain active throughout beacon interval.*

In Superframe order 10 and Beacon order 12 Sample, sleep energy consumed will be non-zero since sensor goes to sleep mode during the inactive portion of the Superframe.

Battery model

Device Name	Initial energy(mJ)	Consumed energ	Remaining Energ	Harvested Energy	Transmitting ener	Receiving energy	Idle energy(mJ)	Sleep energy(mJ)
WIRELESS_SENSOR_	6480.000000	217.020104	6333.539896	70.560000	2.154177	0.655811	185.822992	28.387124

Figure 4-6: *Battery Model Table for SO=10 & BO=12.*

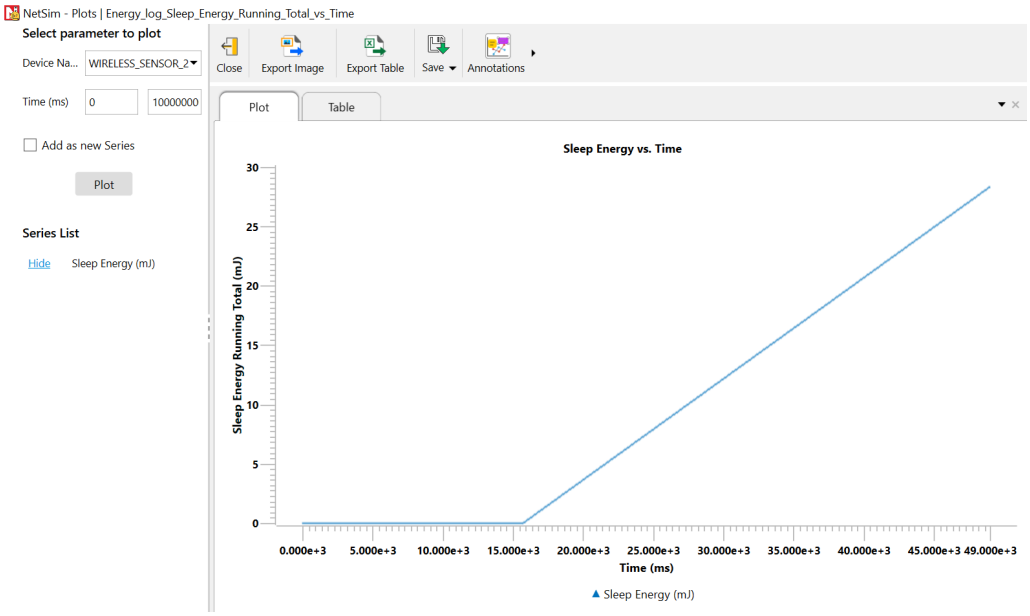


Figure 4-7: *In the above plot, we observe that sleep energy remains zero when SO=10 and BO=12 during the active period, i.e., till 15728.64 ms. After this period, sleep energy consumption increases linearly due to the inactive portion of the super frame.*

4.1.2 Working of RPL Protocol in IoT

Open NetSim, Select Examples ▷IOT-WSN ▷Internet of Things ▷Working of RPL Protocol in IoT then click on the tile in the middle panel to load the example as shown in Figure 4-8.

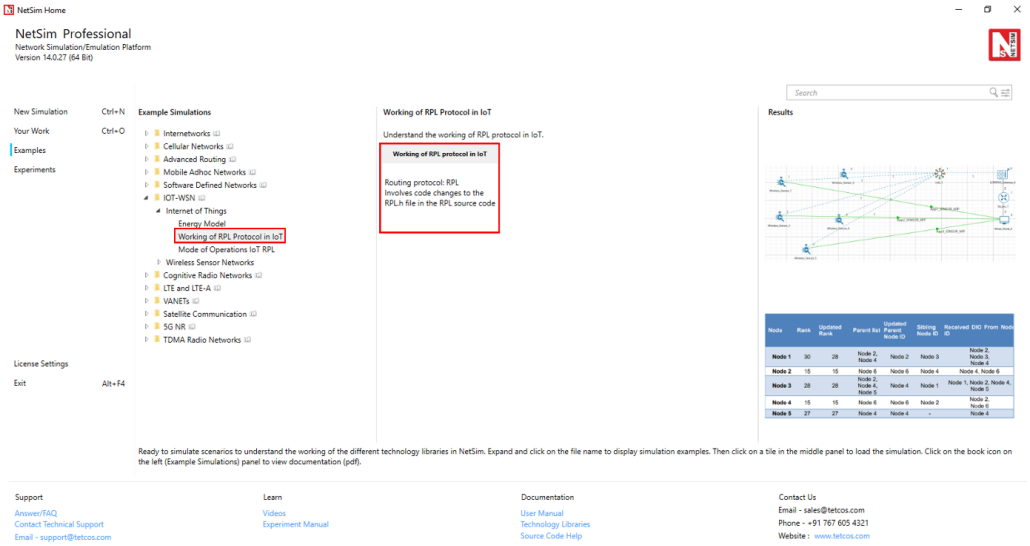


Figure 4-8: *List of scenarios for the example of Working of RPL Protocol in IoT.*

The following network diagram illustrates what the NetSim UI displays when you open the example configuration file.

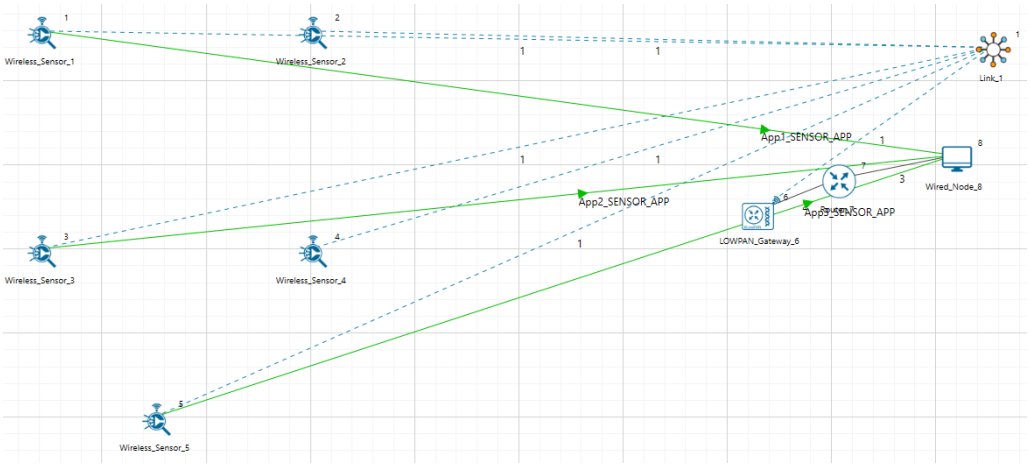


Figure 4-9: *Network set up for studying the Working of RPL Protocol in IoT.*

Settings done in sample network:

- Set grid length as 120 × 60m from grid setting property panel on the right. This needs to be done before any device is placed on the grid.
- Click on LOWPAN Gateway or sensor. In the right property panel, change the routing protocol to RPL under network layer as shown in Figure 4-10.

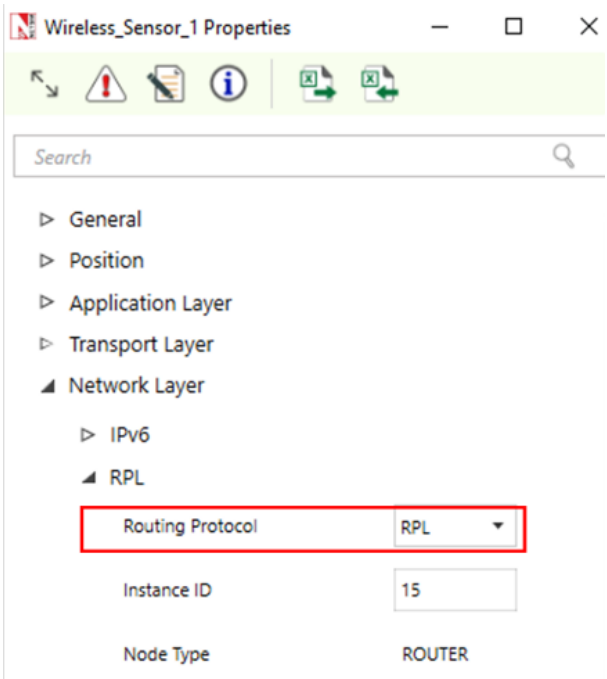


Figure 4-10: *Routing Protocol to RPL in Network layer.*

1. Click on link, expand the right property panel change Channel characteristics → Path Loss only, Path Loss Model → Log Distance and path loss exponent → 3.5.
2. Configure sensor applications as shown in the below table by clicking on set traffic tab from the ribbon on the top.

Table 4-1: *Application properties.*

Application ID	Application Type	Source Id	Destination Id
1	SENSOR APP	1	8
2	SENSOR APP	3	8
3	SENSOR APP	5	8

Procedure to get detailed RPL log file:

- Go to help option in the title bar.
- Click on Open Source code.
- Go to the RPL Project in the Solution Explorer. Open RPL.h file and change `// #define DEBUG_RPL` to `#define DEBUG_RPL` as shown in Figure 4-11.

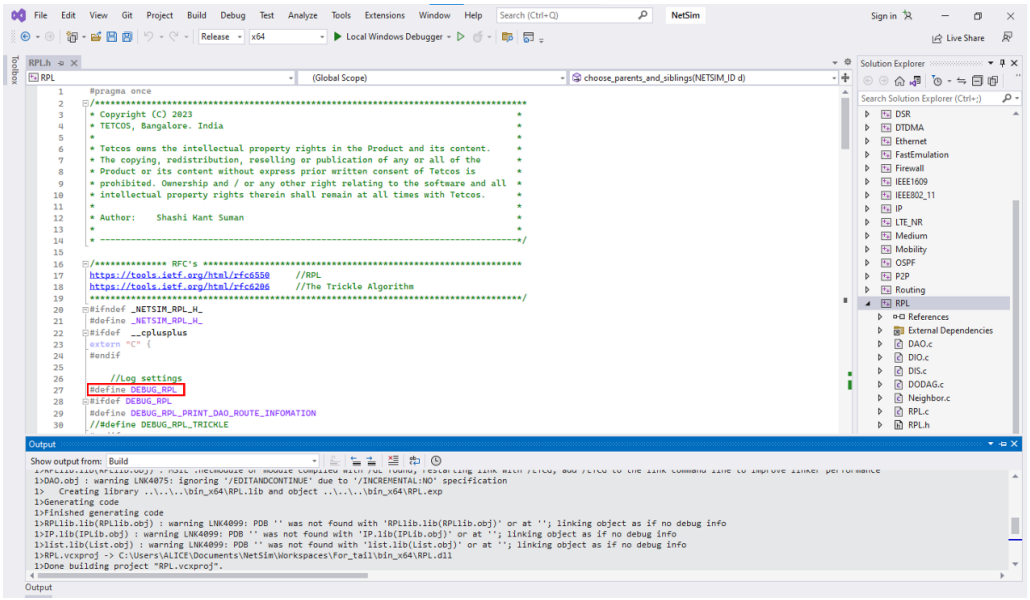


Figure 4-11: *Visual Studio.*

- Right click on the RPL project in the solution explorer and click on rebuild.
- After the RPL project is rebuilt successfully, go back to the network scenario.
- Enable the Application throughput vs time and Link throughput vs time plots under plots in configure reports tab and run simulation for 100 sec and go to Result window → Logs, open rpl log file.

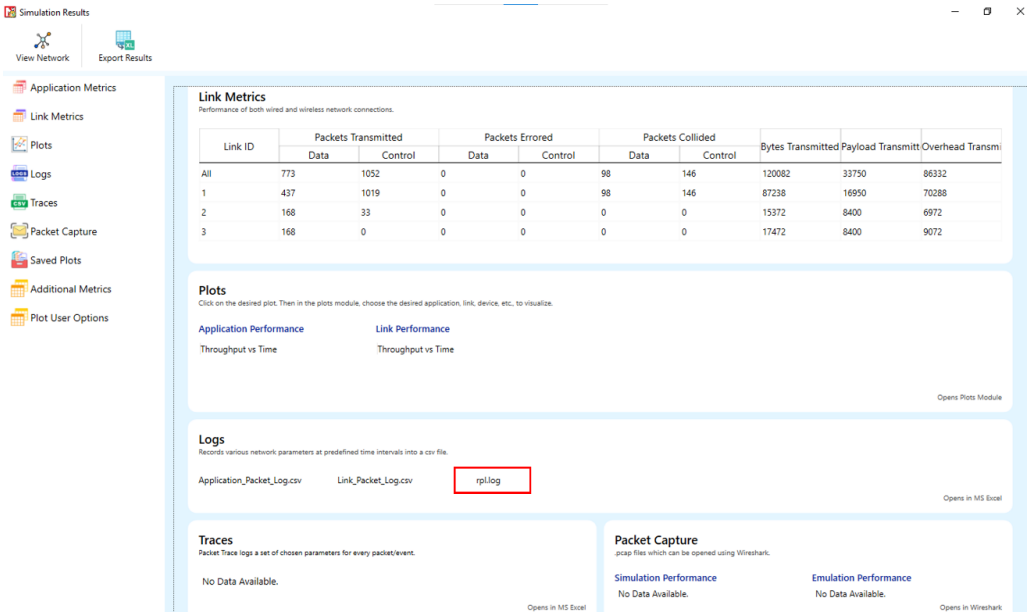


Figure 4-12: *Simulation Result window Results.*

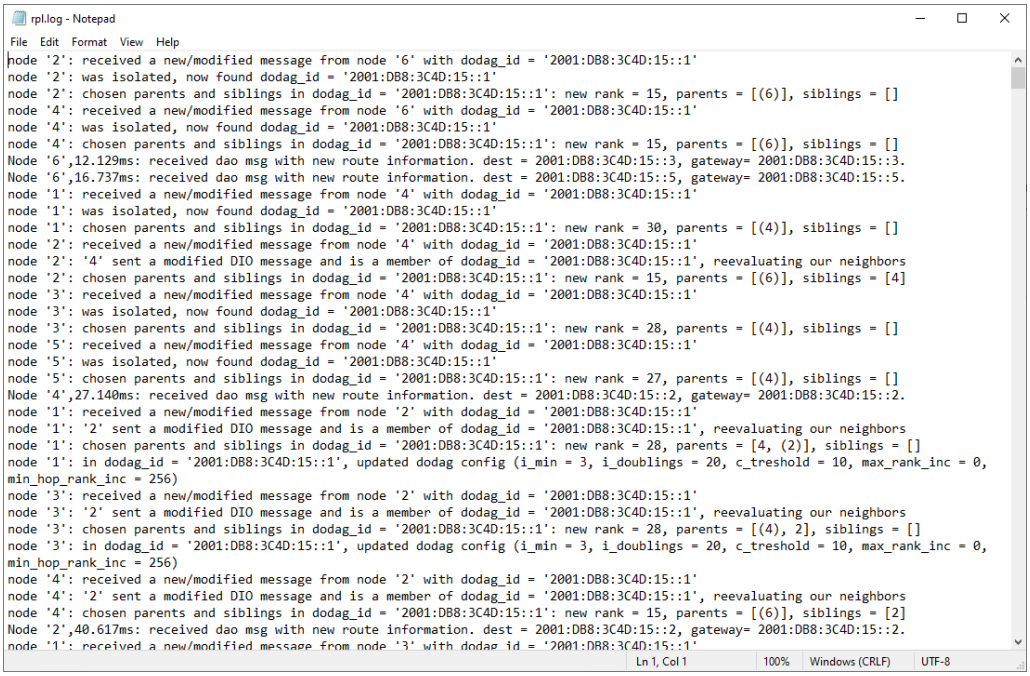


Figure 4-13: *RPL log file.*

Note:

Detailed information of the RPL log file is available for the standard version of NetSim. For academic users, the rank value can be observed by enabling Wireshark and also python script is available to know the parent and sibling information. Please find the below link to download and run the python utility: https://support.tetcos.com/support/solutions/articles/14000134056-how-to-visualize-the-rpl-dodag-in-netsim-iot-simulations-

The observations of rpl log file which is generated in NetSim is given in the below Table 4-2.

Table 4-2: *RPL Log file contains Rank, Updated Rank, parent list, Updated Parent Node ID, Sibling Node ID and Received DIO From Node ID etc.*

Node	Rank	Updated Rank	Parent list	Updated Parent Node ID	Sibling Node ID	Received DIO From Node ID
Node 1	30	27	Node 2, Node 4	Node 2	Node 3	Node 2, Node 3, Node 4
Node 2	15	15	Node 6	Node 6	Node 4	Node 4, Node 6
Node 3	28	28	Node 2, Node 4, Node 5	Node 4	Node 1	Node 1, Node 2, Node 4, Node 5
Node 4	15	15	Node 6	Node 6	Node 2	Node 2, Node 6
Node 5	27	27	Node 4	Node 4	–	Node 4

The above table can be summarized as follows:

DIO message is sent by the root node, i.e. LowPAN Gateway, with Rank 1 to Sensor 2 and Sensor 4 as shown in Figure 4-14.

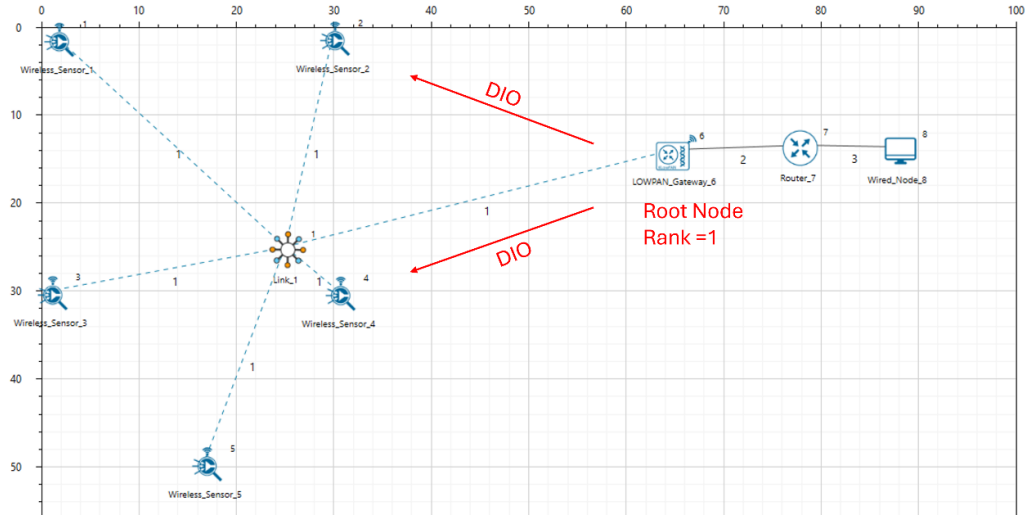


Figure 4-14: *DIO messages sent by the LOWPAN Gateway to Sensors.*

On receiving the DIO message, Node 4 will recognize the DODAG Id of Root Node and identifies it as Parent node. Rank of Node 4 will get updated to 15. Also, Node 2 is recognized as sibling of Node 4.

Now, node 4 will broadcast DIO message to other nodes as shown in Figure 4-15.

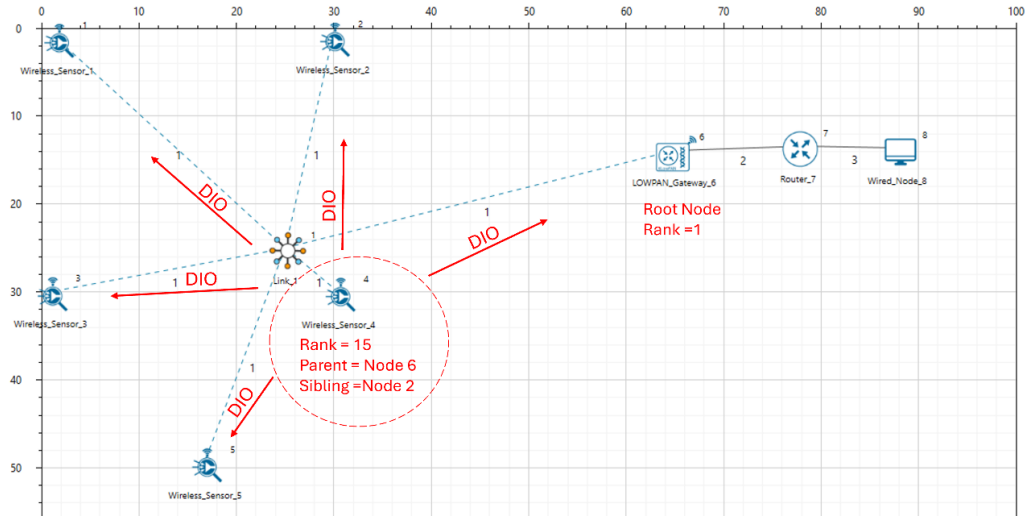


Figure 4-15: *Wireless Sensor 4 broadcasting DIO message to other Sensors.*

Node 1 receives DIO message from Node 4 and it identifies the DODAG Id of Node 4. Hence, Node 1 recognizes Node 4 as the Parent Node. Rank of Node 1 will get updated to 30. As Node 3 is within the range of Node 1, Node 3 is identified as a sibling of Node 1.

Node 1 will then broadcast DIO messages as shown in Figure 4-16.

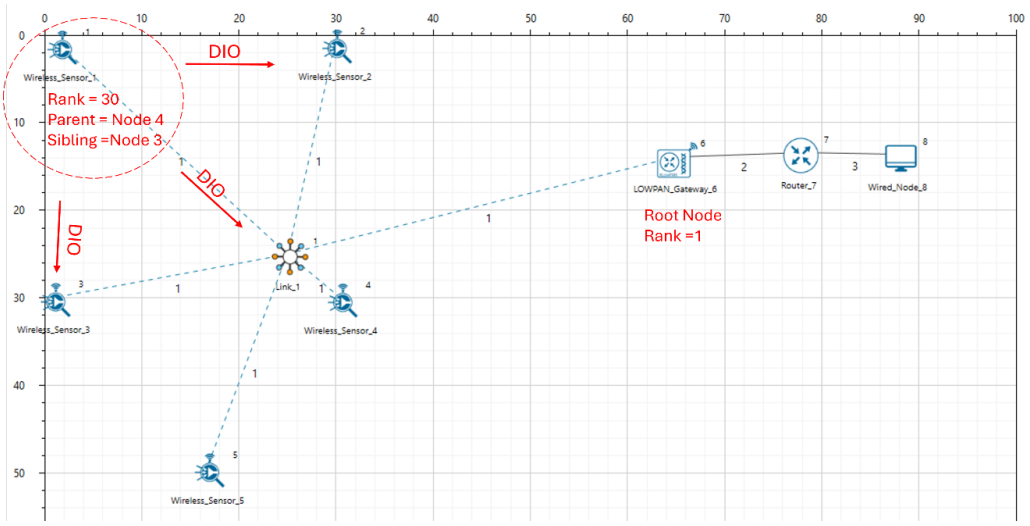


Figure 4-16: *Wireless Sensor 1 broadcasting DIO message to other Sensors.*

Node 2 receives the DIO message from Node 6 and identifies it as Parent node. The Rank of Node 2 gets updated to 15. Node 2 now broadcasts the DIO message to other nodes as shown in Figure 4-17.

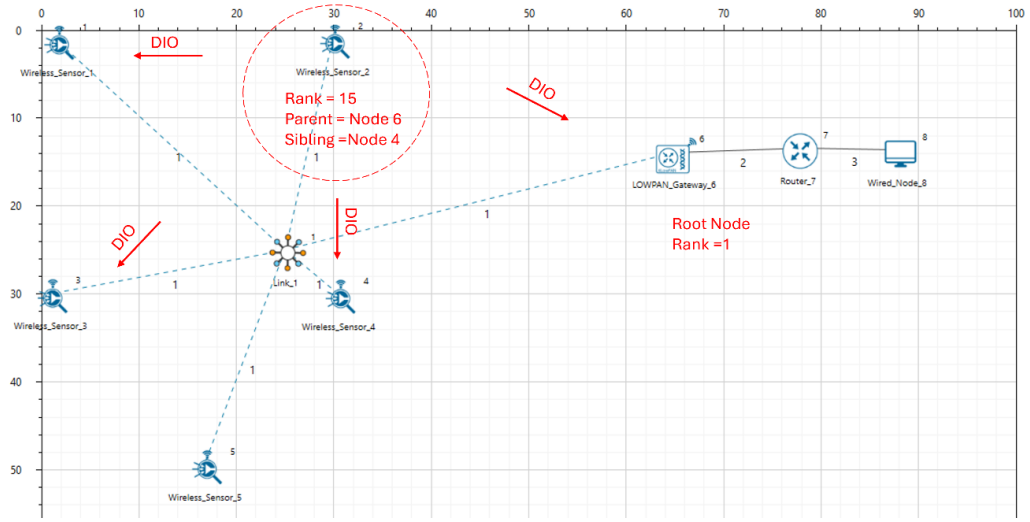


Figure 4-17: *Wireless Sensor 2 broadcasting DIO message to other Sensors.*

Node 3 receives DIO message from Node 4 and the rank of Node 3 gets updated to 28. Node 4 is identified as the parent node of Node 3.

Node 3 then broadcasts DIO message to other nodes as shown in Figure 4-18.

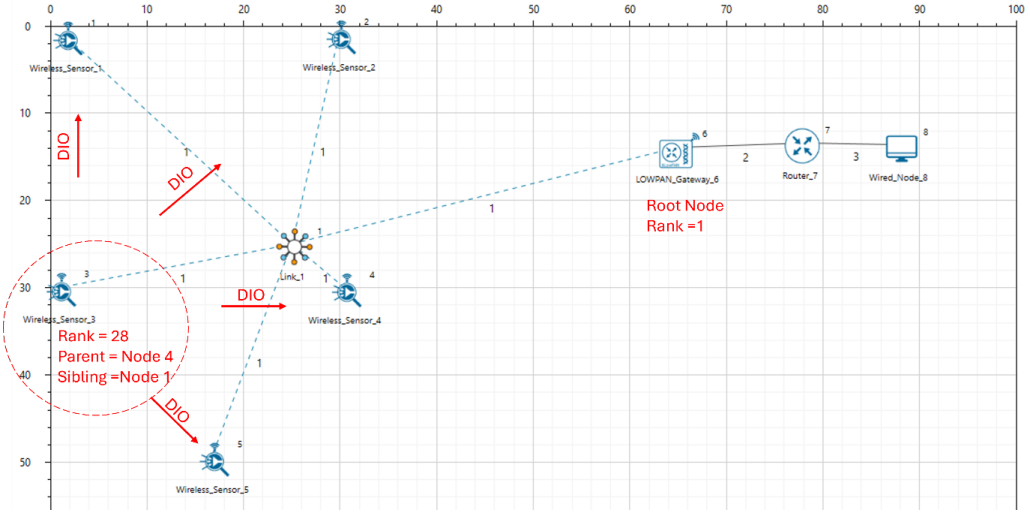


Figure 4-18: *Wireless Sensor 3 broadcasting DIO message to other Sensors.*

Node 5 receives DIO message from Node 3, hence, it identifies Node 4 as the parent node. The rank of Node 5 gets updated to 27.

Node 5 then broadcasts DIO message as shown in Figure 4-19.

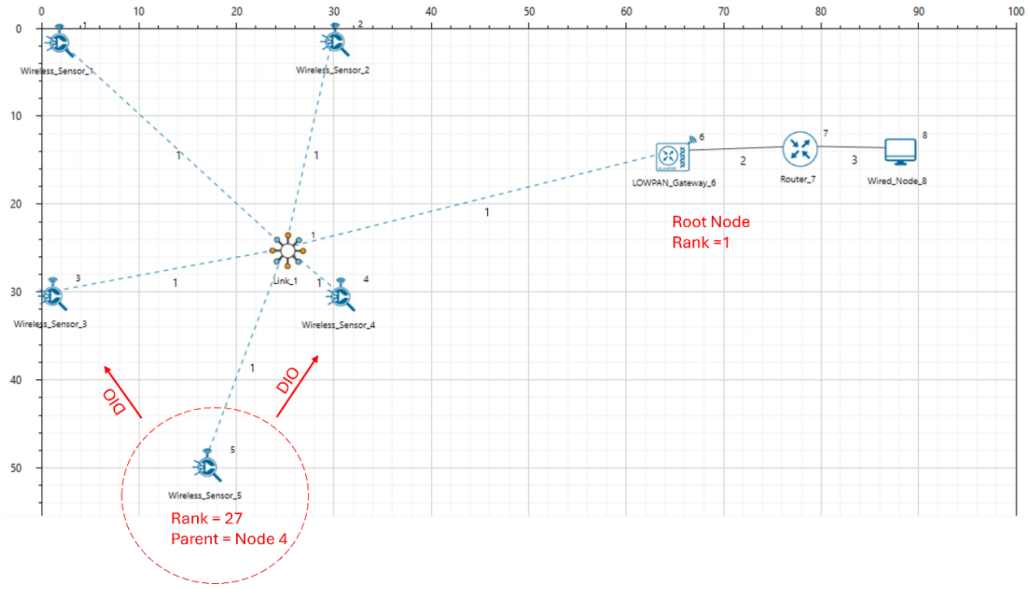


Figure 4-19: *Wireless Sensor 5 broadcasting DIO message to other Sensors.*

Further, Node 1 receives DIO message from Node 2 and the parent list of Node 1 gets updated to Node 4 and Node 2. Also, the rank of Node 1 gets updated to 27.

According to the link quality, DODAG is formed.

- Node 2 and Node 4 are siblings and their parent is Node 6 (Root Node). Rank is 15.
- Node 1 and Node 3 are siblings.
  - Node 1 establishes its parent as Node 2. Rank is 27.
  - Node 3 establishes its parent as Node 4. Rank is 28.
- Node 5 doesn't have any siblings and establishes its parent as Node 4. Its rank is 27.

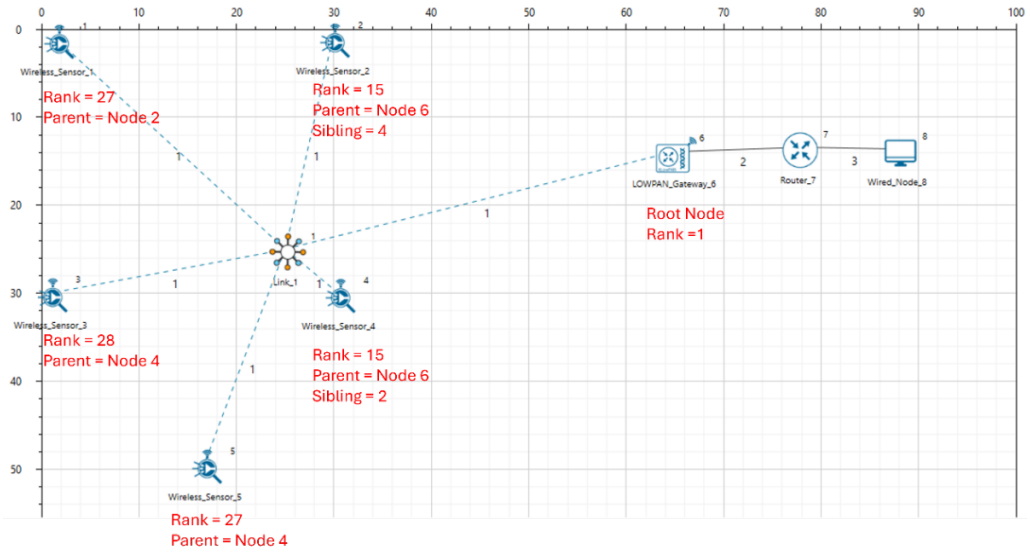


Figure 4-20: *Based on link quality, DODAG is formed and Rank assigned to sensors and lowpan gateway.*

4.1.3 Modes of Operation in IoT RPL

The DODAG nodes process the DAO messages according to the RPL Mode of Operations (MOP), which are presented below. Independent of the MOP used, DAO messages may require reception confirmation, which should be done using DAO-ACK messages.

Although it is designed for the Multipoint-to-Point (MP2P) traffic pattern, RPL also admits the data forwarding using Point-to-Multipoint (P2MP) and Point-to-Point (P2P). In MP2P, the nodes send data messages to the root, creating an upward flow as shown in Figure 4-21. In P2MP, sometimes termed as multicast, the root sends data messages to the other nodes, producing a downward flow depicted in Figure 4-23.

In P2P, a node sends messages to the other node (non-root) of DODAG; thus, both upward and downward forwarding may be required as illustrated in Figure 4-22. RPL defines four MOPs that should be used considering the traffic pattern required by the application and the computational capacity of the nodes. In the first, MOP 0 (Point to multipoint), RPL does not maintain downward routes; thus, consequently, only MP2P traffic is enabled. In storing without multicast MOP (MOP 2 (Point to point)), downward routes are also supported, but are different from MOP 1; the nodes maintain, individually, a routing table constructed using DAO messages to provide downward traffic. Hence, downward forwarding occurs without the use of the root node, as illustrated in Figure 4-24. Storing with multicast has a functioning similar to MOP 2 (Point to point) plus the possibility of multicast data sending. This type of transmission permits the non-root node to send messages to a group of nodes formed using multicast DAOs.

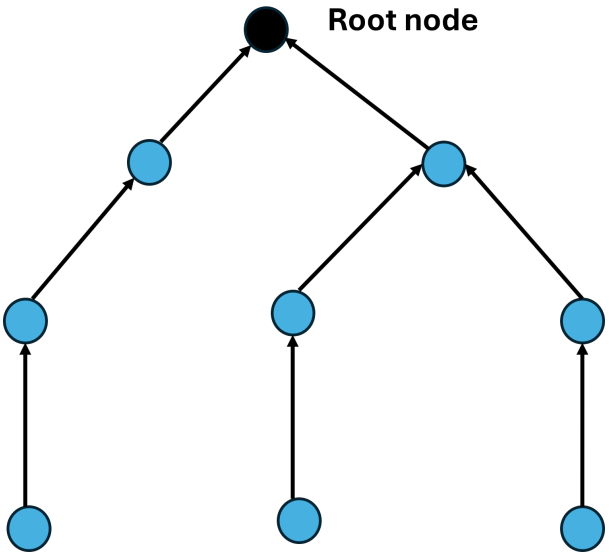


Figure 4-21: *Multipoint to Point.*

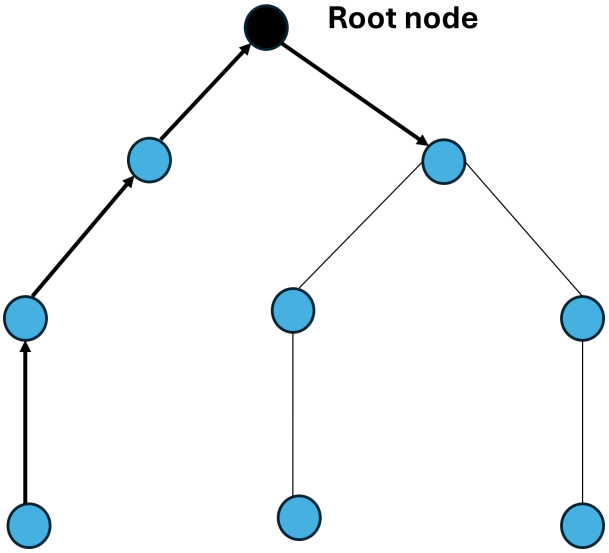


Figure 4-22: *Point to Point.*

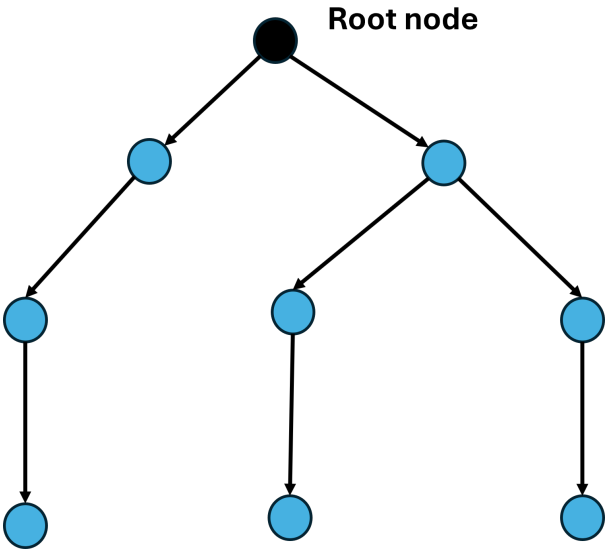


Figure 4-23: *Point to Multipoint.*

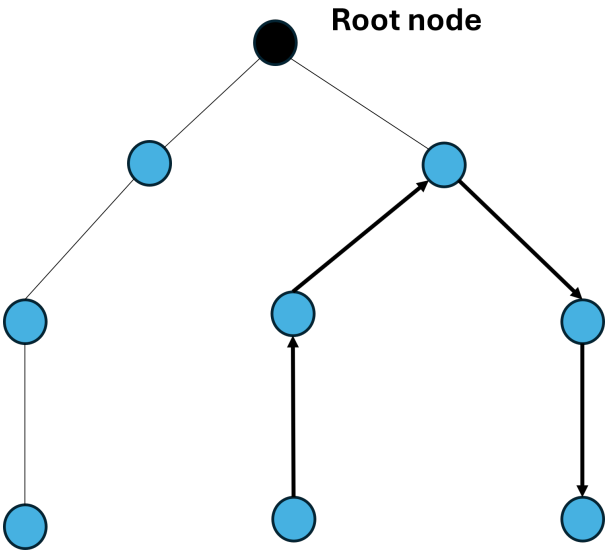


Figure 4-24: *RPL Storing Mode.*

Open NetSim, Select Examples ▷IOT-WSN ▷Internet of Things ▷Mode of Operations IoT RPL then click on the tile in the middle panel to load the example as shown in Figure 4-25.

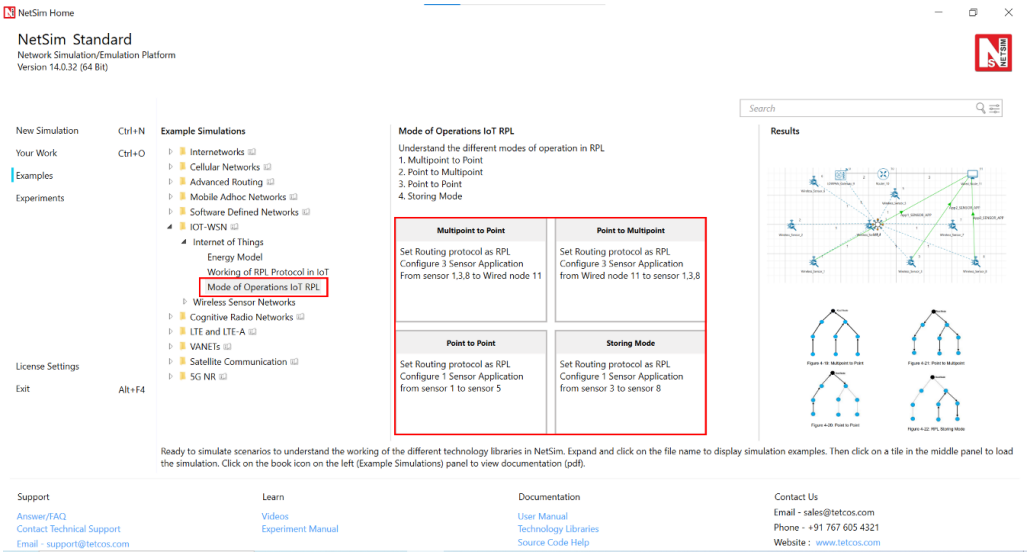


Figure 4-25: *List of scenarios for the example of Mode of Operations IoT RPL.*

The following network diagram illustrates what the NetSim UI displays when you open the example configuration file.

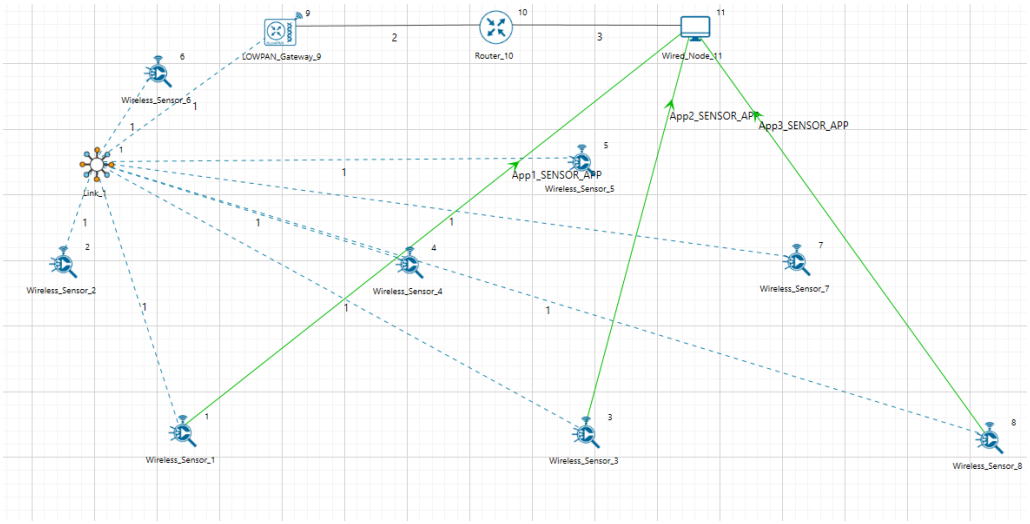


Figure 4-26: *Network set up for studying the Mode of Operations IoT RPL.*

Multipoint to Point

Settings done in sample network:

1. Set grid length as 200 × 100m from grid setting property panel on the right. This needs to be done before any device is placed on the grid.
2. Routing protocol has been set as RPL for LOWPAN Gateway and Sensor. Click on device, expand the right-side property panel and go to Network Layer set Routing Protocol as shown in Figure 4-27.

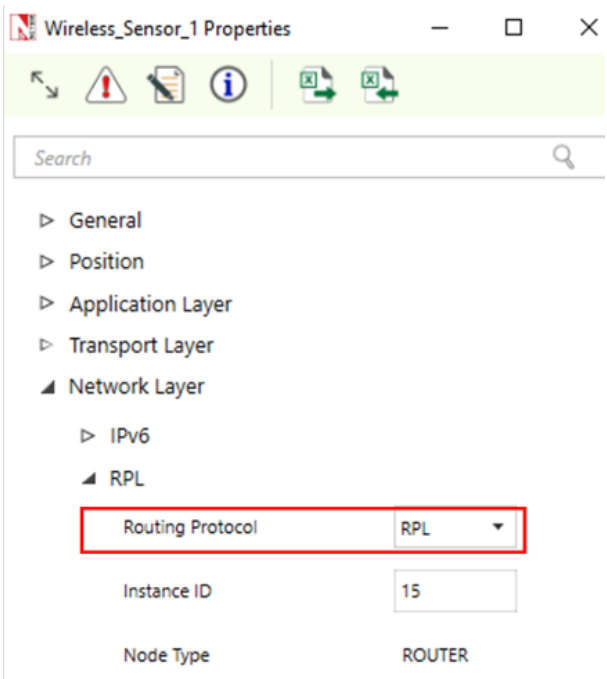


Figure 4-27: *Routing Protocol to RPL in Network layer.*

3. Click on the link, expand the right-side property panel, and set the channel characteristics to 'Path Loss Only'. Then, set the path loss model to 'Log Distance' and the path loss exponent to 4.2.
4. Configure sensor applications as shown in the below table by clicking on set traffic tab from the ribbon on the top.

Table 4-3: *Application properties.*

Application ID	Application Type	Source Id	Destination Id
1	SENSOR APP	1	11
2	SENSOR APP	3	11
3	SENSOR APP	8	11

4. In NetSim GUI Packet Traces and Plots are Enabled. Run simulation 100s and observe that all the sensors are sending data to the root node in packet trace.

Result

Packet trace: The nodes send data messages to the root, creating an upward flow which can be observed in the packet trace. Once the simulation is completed, to view the packet trace file, click on "Packet Trace" under Traces option present in the left-hand-side of the Simulation Results Dashboard and filter the PACKET TYPE to Sensing. You can observe the flow of Sensor Application packets as shown in Figure 4-28.

	PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
131	2	0	Sensing	App3_SENSOR_APP	SENSOR-8	NODE-11	SENSOR-8	SENSOR-7
132	2	0	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-11	SENSOR-1	SENSOR-2
133	2	0	Sensing	App3_SENSOR_APP	SENSOR-8	NODE-11	SENSOR-7	SENSOR-5
134	2	0	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-11	SENSOR-2	SENSOR-6
135	2	0	Sensing	App2_SENSOR_APP	SENSOR-3	NODE-11	SENSOR-3	SENSOR-4
138	2	0	Sensing	App3_SENSOR_APP	SENSOR-8	NODE-11	SENSOR-5	SINKNODE-9
139	2	0	Sensing	App3_SENSOR_APP	SENSOR-8	NODE-11	SINKNODE-9	ROUTER-10
140	2	0	Sensing	App3_SENSOR_APP	SENSOR-8	NODE-11	ROUTER-10	NODE-11
164	3	0	Sensing	App2_SENSOR_APP	SENSOR-3	NODE-11	SENSOR-3	SENSOR-4
165	3	0	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-11	SENSOR-1	SENSOR-2
166	3	0	Sensing	App3_SENSOR_APP	SENSOR-8	NODE-11	SENSOR-8	SENSOR-7
167	3	0	Sensing	App2_SENSOR_APP	SENSOR-3	NODE-11	SENSOR-4	SENSOR-5
168	3	0	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-11	SENSOR-2	SENSOR-6
179	4	0	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-11	SENSOR-1	SENSOR-2
180	4	0	Sensing	App3_SENSOR_APP	SENSOR-8	NODE-11	SENSOR-8	SENSOR-7
181	4	0	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-11	SENSOR-2	SENSOR-6
182	4	0	Sensing	App2_SENSOR_APP	SENSOR-3	NODE-11	SENSOR-3	SENSOR-4
183	4	0	Sensing	App3_SENSOR_APP	SENSOR-8	NODE-11	SENSOR-7	SENSOR-5
184	4	0	Sensing	App2_SENSOR_APP	SENSOR-3	NODE-11	SENSOR-4	SENSOR-5
186	4	0	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-11	SENSOR-6	SINKNODE-9
211	5	0	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-11	SENSOR-1	SENSOR-2
212	5	0	Sensing	App3_SENSOR_APP	SENSOR-8	NODE-11	SENSOR-8	SENSOR-7
213	5	0	Sensing	App2_SENSOR_APP	SENSOR-3	NODE-11	SENSOR-3	SENSOR-4
214	5	0	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-11	SENSOR-2	SENSOR-6
215	5	0	Sensing	App3_SENSOR_APP	SENSOR-8	NODE-11	SENSOR-7	SENSOR-5
216	5	0	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-11	SENSOR-6	SINKNODE-9
217	5	0	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-11	SINKNODE-9	ROUTER-10
218	5	0	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-11	ROUTER-10	NODE-11

Figure 4-28: *Packet Trace for Multipoint to Point.*

Point to Multipoint

Settings done in sample network:

1. Set all the properties the same as Multipoint to Point scenario.
2. Configure sensor applications as shown in the below table by clicking on set traffic tab from the ribbon on the top.

Table 4-4: *Application properties.*

Application ID	Application Type	Source Id	Destination Id
1	SENSOR APP	11	1
2	SENSOR APP	11	3
3	SENSOR APP	11	8

3. Run simulation 100s and observe that all the sensors are sending data to the root node in packet trace.

Result

Packet trace: Root sends data messages to the other nodes, producing a downward flow which can be observed in the packet trace. Once the simulation is completed, to view the packet trace file, click on "Packet Trace" under Traces option present in the left-hand-side of the Simulation Results Dashboard and filter the PACKET TYPE to Sensing, you can observe the flow of Sensor Application packets as shown below:

PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
1	0	Sensing	App1_SENSOR_APP	NODE-11	SENSOR-1	NODE-11	ROUTER-10
1	0	Sensing	App2_SENSOR_APP	NODE-11	SENSOR-3	NODE-11	ROUTER-10
1	0	Sensing	App1_SENSOR_APP	NODE-11	SENSOR-1	ROUTER-10	SINKNODE-9
1	0	Sensing	App3_SENSOR_APP	NODE-11	SENSOR-8	NODE-11	ROUTER-10
1	0	Sensing	App2_SENSOR_APP	NODE-11	SENSOR-3	ROUTER-10	SINKNODE-9
1	0	Sensing	App3_SENSOR_APP	NODE-11	SENSOR-8	ROUTER-10	SINKNODE-9
2	0	Sensing	App1_SENSOR_APP	NODE-11	SENSOR-1	NODE-11	ROUTER-10
2	0	Sensing	App2_SENSOR_APP	NODE-11	SENSOR-3	NODE-11	ROUTER-10
2	0	Sensing	App1_SENSOR_APP	NODE-11	SENSOR-1	ROUTER-10	SINKNODE-9
2	0	Sensing	App3_SENSOR_APP	NODE-11	SENSOR-8	NODE-11	ROUTER-10
2	0	Sensing	App2_SENSOR_APP	NODE-11	SENSOR-3	ROUTER-10	SINKNODE-9
2	0	Sensing	App3_SENSOR_APP	NODE-11	SENSOR-8	ROUTER-10	SINKNODE-9
3	0	Sensing	App1_SENSOR_APP	NODE-11	SENSOR-1	NODE-11	ROUTER-10
3	0	Sensing	App2_SENSOR_APP	NODE-11	SENSOR-3	NODE-11	ROUTER-10
3	0	Sensing	App1_SENSOR_APP	NODE-11	SENSOR-1	ROUTER-10	SINKNODE-9
3	0	Sensing	App3_SENSOR_APP	NODE-11	SENSOR-8	NODE-11	ROUTER-10
3	0	Sensing	App2_SENSOR_APP	NODE-11	SENSOR-3	ROUTER-10	SINKNODE-9
3	0	Sensing	App3_SENSOR_APP	NODE-11	SENSOR-8	ROUTER-10	SINKNODE-9
4	0	Sensing	App1_SENSOR_APP	NODE-11	SENSOR-1	NODE-11	ROUTER-10
4	0	Sensing	App2_SENSOR_APP	NODE-11	SENSOR-3	NODE-11	ROUTER-10
4	0	Sensing	App1_SENSOR_APP	NODE-11	SENSOR-1	ROUTER-10	SINKNODE-9
4	0	Sensing	App3_SENSOR_APP	NODE-11	SENSOR-8	NODE-11	ROUTER-10
4	0	Sensing	App2_SENSOR_APP	NODE-11	SENSOR-3	ROUTER-10	SINKNODE-9
4	0	Sensing	App3_SENSOR_APP	NODE-11	SENSOR-8	ROUTER-10	SINKNODE-9
5	0	Sensing	App1_SENSOR_APP	NODE-11	SENSOR-1	NODE-11	ROUTER-10
5	0	Sensing	App2_SENSOR_APP	NODE-11	SENSOR-3	NODE-11	ROUTER-10
5	0	Sensing	App1_SENSOR_APP	NODE-11	SENSOR-1	ROUTER-10	SINKNODE-9
5	0	Sensing	App3_SENSOR_APP	NODE-11	SENSOR-8	NODE-11	ROUTER-10

Figure 4-29: *Packet trace for Point to Multipoint.*

Point to Point

Settings done in sample network:

1. Set all the properties the same as Multipoint to Point scenario.
2. Configure sensor applications as shown in the below table by clicking on set traffic tab from the ribbon on the top.

Table 4-5: *Application properties.*

Application ID	Application Type	Source Id	Destination Id
1	SENSOR APP	1	5

2. Run simulation 100s and observe that all the sensors are sending data to the root node in packet trace.

Result

Packet trace: Root sends data messages to the other nodes, producing a downward flow which can be observed in the packet trace. Once the simulation is completed, to view the packet trace file, click on "Packet Trace" under Traces option present in the left-hand-side of the Simulation Results Dashboard and filter the PACKET TYPE to Sensing, you can observe the flow of Sensor Application packets as shown below:

1	PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
131	2	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-1	SENSOR-2
132	2	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-2	SENSOR-6
135	2	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-6	SINKNODE-9
137	2	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SINKNODE-9	SENSOR-5
159	3	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-1	SENSOR-2
160	3	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-2	SENSOR-6
163	3	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-6	SINKNODE-9
166	3	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SINKNODE-9	SENSOR-5
174	4	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-1	SENSOR-2
175	4	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-2	SENSOR-6
177	4	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-6	SINKNODE-9
178	4	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SINKNODE-9	SENSOR-5
202	5	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-1	SENSOR-2
203	5	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-2	SENSOR-6
205	5	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-6	SINKNODE-9
214	6	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-1	SENSOR-2
215	6	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-2	SENSOR-6
217	6	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-6	SINKNODE-9
225	7	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-1	SENSOR-2
226	7	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-2	SENSOR-6
228	7	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-6	SINKNODE-9
246	8	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-1	SENSOR-2
247	8	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-2	SENSOR-6
250	8	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-6	SINKNODE-9
251	8	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SINKNODE-9	SENSOR-5
265	9	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-1	SENSOR-2
266	9	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-2	SENSOR-6
268	9	0	Sensing	App1_SENSOR_APP	SENSOR-1	SENSOR-5	SENSOR-6	SINKNODE-9

Figure 4-30: *Packet trace for Point to Point.*

Storing Mode

Settings done in sample network:

1. Set all the properties the same as Multipoint to Point scenario.
2. Configure sensor applications as shown in the below table by clicking on set traffic tab from the ribbon on the top.

Table 4-6: *Application properties.*

Application ID	Application Type	Source Id	Destination Id
1	SENSOR APP	3	8

3. Run simulation 100s and observe that all the sensors are sending data to root node in packet trace.

Result

In storing without multicast MOP (MOP 2 (Point to point)), downward routes are also supported, but are different from MOP 1 (Point to multipoint); the nodes maintain, individually, a routing table constructed using DAO messages to provide downward traffic. Hence, downward forwarding occurs without the use of the root node. The results can be analyzed from the packet trace.

Packet trace: Once the simulation is completed, to view the packet trace file, click on “Packet Trace” under Traces option present in the left-hand-side of the Simulation Results Dashboard and filter the PACKET TYPE to Sensing, you can observe the flow of Sensor Application packets as shown below.

	PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
131	2	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-3	SENSOR-4
132	2	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-4	SENSOR-5
134	2	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-5	SINKNODE-9
158	3	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-3	SENSOR-4
159	3	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-4	SENSOR-5
161	3	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-5	SENSOR-7
163	3	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-7	SENSOR-8
172	4	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-3	SENSOR-4
173	4	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-4	SENSOR-5
175	4	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-5	SENSOR-7
178	4	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-7	SENSOR-8
201	5	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-3	SENSOR-4
202	5	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-4	SENSOR-5
204	5	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-5	SENSOR-7
209	5	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-7	SENSOR-8
214	6	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-3	SENSOR-4
215	6	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-4	SENSOR-5
217	6	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-5	SENSOR-7
219	6	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-7	SENSOR-8
226	7	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-3	SENSOR-4
227	7	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-4	SENSOR-5
229	7	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-5	SENSOR-7
232	7	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-7	SENSOR-8
249	8	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-3	SENSOR-4
250	8	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-4	SENSOR-5
252	8	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-5	SENSOR-7
253	8	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-7	SENSOR-8
269	9	0	Sensing	App1_SENSOR_APP	SENSOR-3	SENSOR-8	SENSOR-3	SENSOR-4

Figure 4-31: *Packet trace for RPL Storing Mode.*

4.1.4 DDoS attack in an IoT Network

Introduction A DoS (Denial of Service) attack in IoT (Internet of Things) involves overwhelming a device or network with traffic to disrupt its normal operation and deny service to legitimate users. This can be achieved by flooding the network with a large volume of traffic or sending packets with malformed headers to exploit vulnerabilities in the target system. The impact of a DoS attack on an IoT device or network can be significant, as many IoT devices have limited processing power and memory resources. As a result, they may not be able to handle the large volume of traffic generated by the attack, leading to service disruptions or even device failures. When multiple attackers coordinate a DoS attack, it is known as a DDoS (Distributed Denial of Service) attack.

Botnet Attack A botnet attack in IoT (Internet of Things) is a type of cyber-attack where a network of compromised IoT devices, such as smart home appliances, security cameras, and routers, is used to carry out malicious activities. In a botnet attack, a hacker gains control of a large number of IoT devices, often through exploiting vulnerabilities or weak security measures, and uses them to launch coordinated attacks.

Once a hacker has control of a botnet, they can carry out a variety of malicious activities, including:

- DDoS attacks: A distributed denial-of-service (DDoS) attack floods a website or network with traffic, making it unavailable to legitimate users.
- Data theft: A hacker can use a botnet to steal sensitive data, such as financial information or personal data, from devices on the network.
- Spamming: A botnet can be used to send large volumes of spam emails or messages.
- Crypto mining: A hacker can use a botnet to mine cryptocurrency using the processing power of compromised devices.

Bit-and-piece Attack A "bit and piece" DDoS attack in an IoT network is a type of Distributed Denial of Service (DDoS) attack that targets Internet of Things (IoT) devices by sending small amounts of traffic to a large number of devices. This type of attack is also known as a low-rate DDoS attack.

In a bit-and-piece attack, the attacker sends a small amount of traffic to multiple IoT devices, such as smart home appliances, security cameras, or routers. The traffic may include a range of packet types, such as TCP, UDP, and ICMP, and the packets may be sent intermittently over a long period of time. Because

the traffic is distributed across many devices, each individual device receives a small amount of traffic that is difficult to distinguish from legitimate traffic. However, when combined, the traffic overwhelms the network and causes a denial of service for legitimate users trying to access the network.

Network Setup for Botnet Attack Case 1: Without Malicious Node (No attacker)

In NetSim, click on Examples ▷ IOT-WSN ▷ DDoS attack in an IOT Network

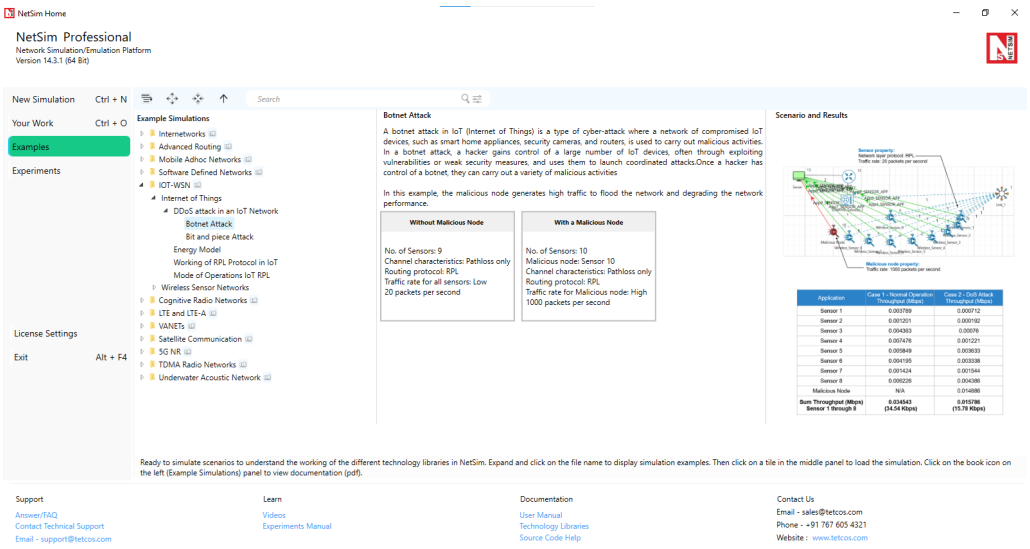


Figure 4-32: *List of scenarios for the example of DDoS attack in an IOT Network.*

Drop 9 Sensors, 1 LOWPAN gateway, 1 router, and 1 wired node(Server) as shown in the figure.

Click on the sensor, expand the property panel on right and set the routing protocol to RPL.

Click on the link and expand the link property panel on the right. Set the channel characteristics to Path Loss only, Path Loss Model to Log Distance, and path loss exponent to 4.5.

Configure sensor traffic from all sensors to server by clicking on set traffic tab in ribbon on top, such that the packets are transmitted to the gateway via sensor 9. The traffic rate is low; it is 20 packets per second. Set Transport Protocol to UDP, Packet size = 50 Bytes and IAT (μs) = 50000

Run the simulation for 100 seconds and measure the throughput obtained by sensors 1 through 8.

Case 2: With a Malicious Node (DoS attack node)

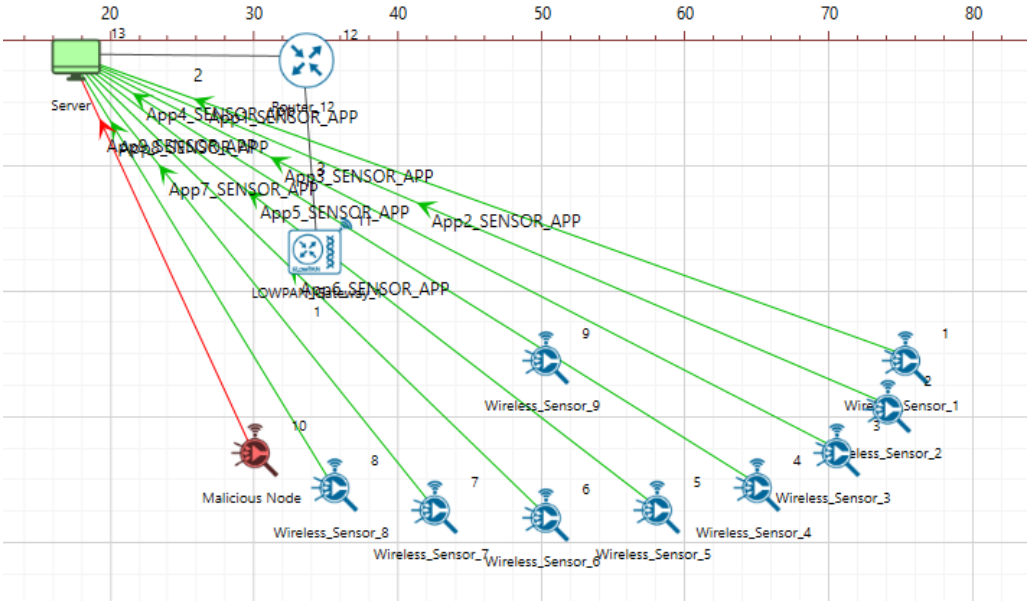


Figure 4-33: *Network setup for 1 malicious node.*

Configure traffic from all sensors 1 through 8 such that the packets are transmitted to the gateway via sensor 9. The traffic rate is low; it is 20 packets per second. Set Transport Protocol to UDP, Packet size = 50 Bytes and IAT (μs) = 50000

Add a malicious node. Configure traffic such that packets are transferred via sensor 9. This has a "high" generation rate to flood the network; it is 1000 packets per second. Set Packet size = 50 Bytes and IAT (μs) = 1000

Run the simulation for 100 seconds and measure the throughput obtained by sensors 1 through 8.

Results and Observation

Table 4-7: *Application Throughput for Case1 and Case 2.*

Application	Case 1 – Normal Operation Throughput (Mbps)	Case 2 – DoS Attack Throughput (Mbps)
Sensor 1	0.003789	0.000712
Sensor 2	0.001201	0.000192
Sensor 3	0.004383	0.00076
Sensor 4	0.007476	0.001221
Sensor 5	0.005849	0.003633
Sensor 6	0.004195	0.003338
Sensor 7	0.001424	0.001544
Sensor 8	0.006226	0.004386
Malicious Node	N/A	0.014886
Sum Throughput (Mbps) Sensor 1 through 8	0.034543 (34.54 Kbps)	0.015786 (15.78 Kbps)

From the table, we observe that the introduction of the malicious node led to a more than 50% drop in throughput, from 34.54 Kbps to 15.78 Kbps.

Network Setup for Bit-and-Piece Attack In NetSim, click on Examples ▷ IOT-WSN ▷ DDoS attack in an IOT Network

Case 1: Without Malicious Node (No attacker)

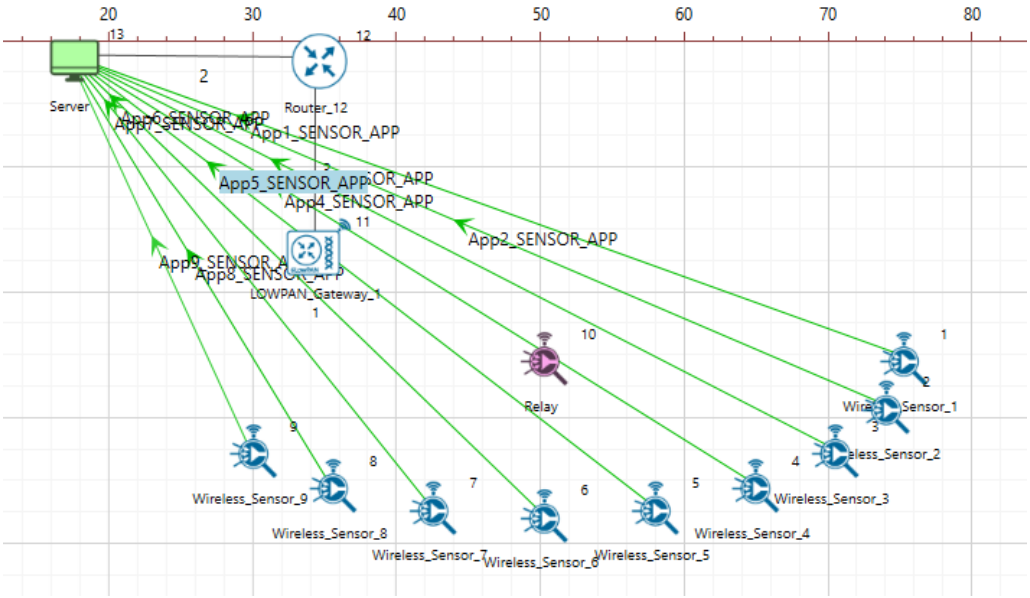


Figure 4-34: *Network setup for normal scenario.*

- Drop 10 sensors, 1 LOWPAN Gateway, 1 Router and 1 Wired node (server) as shown in the figure.
- Click on the sensor, expand the property panel on right and set the routing protocol to RPL.
- Click on the link and expand the link property panel on the right. Set the channel characteristics to Path Loss only, Path Loss Model to Log Distance, and path loss exponent to 4.5.
- Configure 9 applications. Each application is from a sensor (1, 2, ..., 9) to the server, such that packets of size 50 Bytes are sent at a rate of 20 packets/sec. Set Transport Protocol to UDP, Packet size = 50 Bytes and IAT (μs) = 50000
- The network topology and routing are such that traffic from all sensors flows through relay (sensor 10), and then to the gateway and onwards to the server.
- Run the simulation for 100 seconds and measure the throughput obtained by sensors 1 through 9.

We then simulate 3 attack cases.

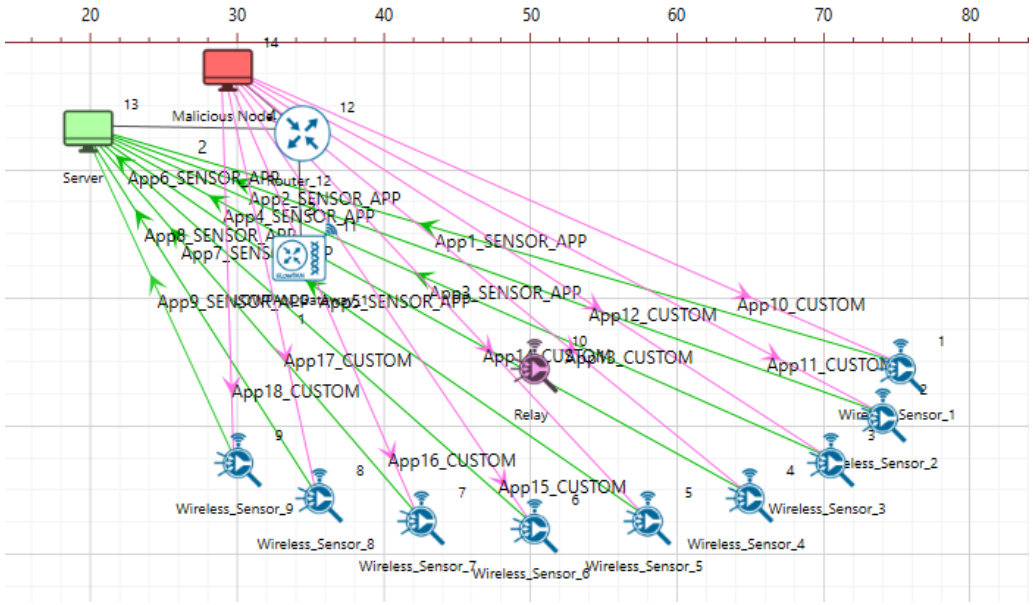


Figure 4-35: *Case 2 – 1 malicious node.*

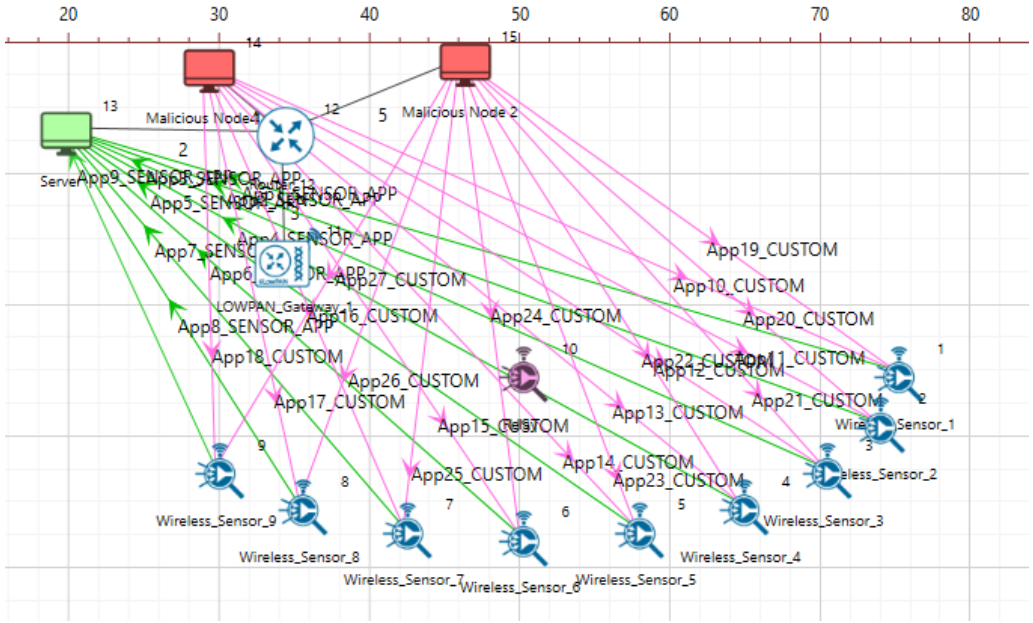


Figure 4-36: *Case 3 – 2 malicious nodes.*

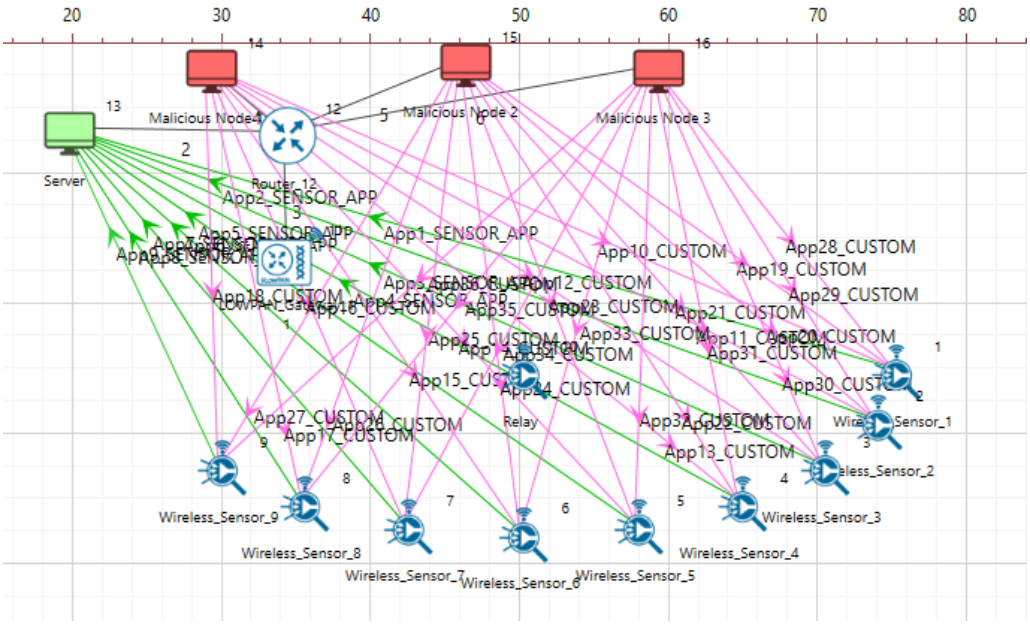


Figure 4-37: *Case 4 – 3 malicious nodes.*

- Case 2: Add 1 malicious node (wired). Configure custom traffic from malicious node to all sensors (1, 2, . . ., 9). Packets of size 10 Bytes (representing a small amount of attack data) are sent at a rate of 20 packets/sec exponentially. Set Transport Protocol to UDP, Packet Size = 10 Bytes and exponential distribution for IAT (μs) = 50000.
- Case 3: Add 2 malicious nodes (wired). Configure custom traffic from 2 malicious nodes to all sensors (1, 2, . . ., 9). Packets of size 10 Bytes (representing a small amount of attack data) are sent at a rate of 20 packets/sec exponentially. Set Transport Protocol to UDP, Packet Size = 10 Bytes and exponential distribution for IAT (μs) = 50000.
- Case 4: Add 3 malicious nodes (wired). Configure custom traffic from 3 malicious nodes to all sensors (1, 2, . . ., 9). Packets of size 10 Bytes (representing a small amount of attack data) are sent at a rate of 20 packets/sec exponentially. Set Transport Protocol to UDP, Packet Size = 10 Bytes and exponential distribution for IAT (μs) = 50000.
- Run the simulation for 100 seconds and measure the throughput obtained by sensors 1 through 9 in each case.

Results and Observation

Table 4-8: *Application Throughput for various cases.*

Application	Case 1: Normal Operation Throughput (Mbps)	Case 2: 1 attacker node Throughput (Mbps)	Case 3: 2 attacker nodes Throughput (Mbps)	Case 4: 3 attacker nodes Throughput (Mbps)
Sensor 1	0.003577	0.002401	0.001968	0.001752
Sensor 2	0.000853	0.000672	0.000596	0.00048
Sensor 3	0.002185	0.001577	0.001445	0.001233
Sensor 4	0.007332	0.004942	0.003966	0.003546
Sensor 5	0.004501	0.003417	0.002724	0.002428
Sensor 6	0.004527	0.00317	0.002582	0.002242
Sensor 7	0.001184	0.000976	0.000788	0.000672
Sensor 8	0.006346	0.004406	0.003609	0.002953
Sensor 9	0.002189	0.001625	0.001329	0.001089
Sum Throughput (Mbps) of Legitimate Traffic	0.032694 (32.6 Kbps)	0.023186 (23.1 Kbps)	0.019007 (19.1 Kbps)	0.016395 (16.3 Kbps)

We observe:

25% drop in throughput of legitimate traffic with 1 bit-and-piece DDoS attack node.

50% drop in throughput of legitimate traffic with 3 bit-and-piece DDoS attack nodes.

4.2 WSN

4.2.1 Beacon Time Analysis

Open NetSim, Select Examples ▷IOT-WSN ▷Wireless Sensor Networks ▷Beacon Time Analysis, then click on the tile in the middle panel to load the example as shown in Figure 4-38.

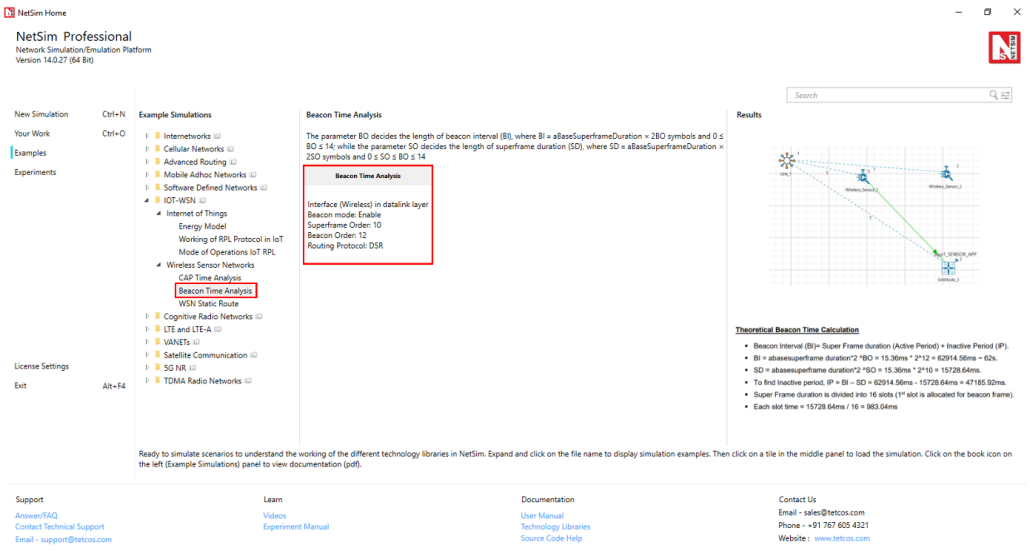


Figure 4-38: *List of scenarios for the example of Beacon Time Analysis.*

The following network diagram illustrates what the NetSim UI displays when you open the example configuration file.

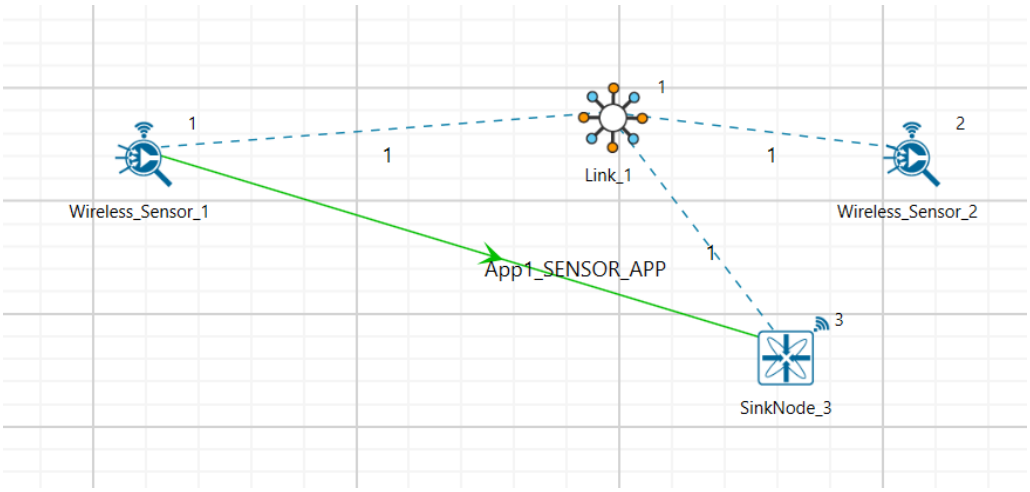


Figure 4-39: *Network set up for studying the Beacon Time Analysis.*

Settings done in sample config file

- The following grid properties are already set, as Manually Via Click and Drop.
- In the SinkNode, the Super frame mode and Beacon mode are set to 10 and 12, respectively. To configure it, click on the SinkNode. On the right side, expand the property panel, go to the data link layer of the Interface (Zigbee) layer, and set the Superframe Order (SO) to 10 and the Beacon Order (BO) to 12.

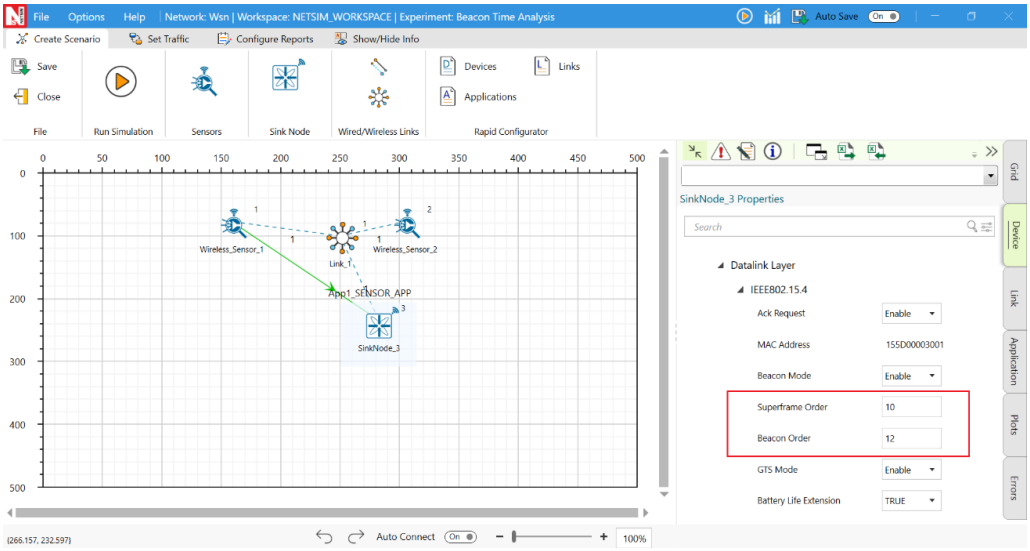


Figure 4-40: *Setting Super frame and Beacon order in Sink node.*

- In the position layer of wireless sensors and the sink, the mobility model is set to Random Waypoint.
- In Adhoc link properties change the channel characteristics as Path Loss only, Path Loss Model to Log Distance and path loss exponent as 2.
- Create sensor traffic between Wireless Sensor 1 and WSN Sink 3 from the Set Traffic tab in the top ribbon. Click on the created application, set the transport layer protocol to UDP, and leave the other properties as default.

- Enable Packet trace from the configure report tab in the ribbon on the top.

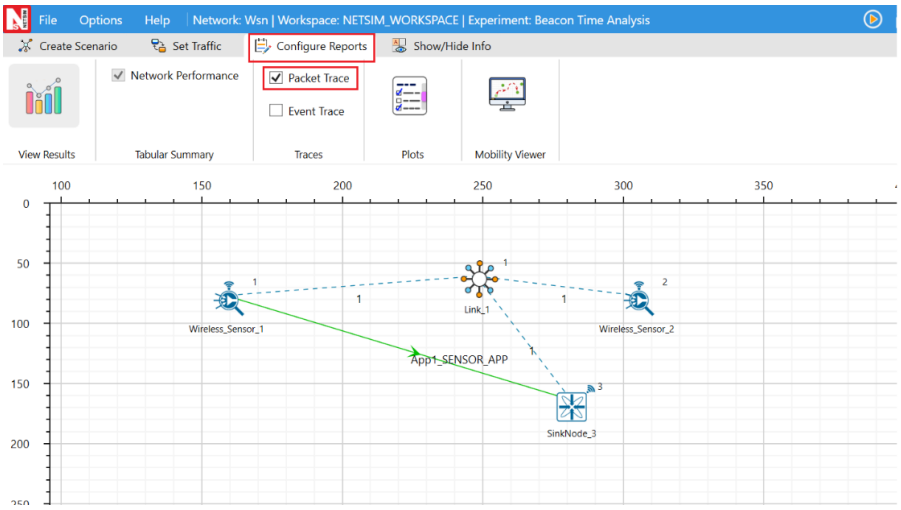


Figure 4-41: *Enabling the Packet trace in Configure Reports tab.*

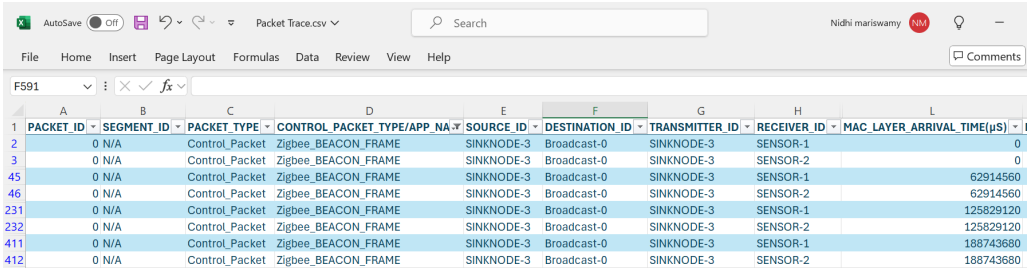
- Run the simulation for 200 seconds.

Theoretical Beacon Time Calculation

- Beacon Interval (BI) = Superframe duration (SD) + Inactive Period (IP). The Superframe duration (SD) is also known as the Active period.
- $BI = aBaseSuperframe\ duration \times 2^{BO} = 15.36ms \times 2^{12} = 62914.56ms \approx 62s.$
- $SD = aBaseSuperframe\ duration \times 2^{SO} = 15.36ms \times 2^{10} = 15728.64ms.$
- $aBaseSuperframe\ duration$ is a constant defined by the standard and is equal to $15.36ms$
- To find Inactive period, $IP = BI - SD = 62914.56ms - 15728.64ms = 47185.92ms.$
- Superframe duration is divided into 16 slots (1st slot is allocated for beacon frame).
- Each slot time $= \frac{15728.64ms}{16} = 983.04ms$

NetSim Results:

- Open packet trace and filter CONTROL PACKET TYPE to Zigbee BEACON FRAME, users should get four Zigbee beacon frames at 0, 62.9, 125.8, 188.7 secs (approx.) for each Sensor Node, because the time interval between two beacon frames is 62 seconds. Since we have 2 nodes, the user can get 4 beacon frames for each node.



PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NA	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	MAC_LAYER_ARRIVAL_TIME(µS)
0	N/A	Control_Packet	Zigbee_BEACON_FRAME	SINKNODE-3	Broadcast-0	SINKNODE-3	SENSOR-1	0
0	N/A	Control_Packet	Zigbee_BEACON_FRAME	SINKNODE-3	Broadcast-0	SINKNODE-3	SENSOR-2	0
0	N/A	Control_Packet	Zigbee_BEACON_FRAME	SINKNODE-3	Broadcast-0	SINKNODE-3	SENSOR-1	62914560
0	N/A	Control_Packet	Zigbee_BEACON_FRAME	SINKNODE-3	Broadcast-0	SINKNODE-3	SENSOR-2	62914560
0	N/A	Control_Packet	Zigbee_BEACON_FRAME	SINKNODE-3	Broadcast-0	SINKNODE-3	SENSOR-1	125829120
0	N/A	Control_Packet	Zigbee_BEACON_FRAME	SINKNODE-3	Broadcast-0	SINKNODE-3	SENSOR-2	125829120
0	N/A	Control_Packet	Zigbee_BEACON_FRAME	SINKNODE-3	Broadcast-0	SINKNODE-3	SENSOR-1	188743680
0	N/A	Control_Packet	Zigbee_BEACON_FRAME	SINKNODE-3	Broadcast-0	SINKNODE-3	SENSOR-2	188743680

Figure 4-42: *Packet Trace showing the Zigbee beacon frames for each sensor node.*

- 4 beacons were transmitted, so *beacon time* $= 983.04ms \times 4 = 3932.16$ *millisec* (since 1 beacon = 1 time slot). Check the beacon time in IEEE 802.15.4 metrics window.

4.2.2 CAP Time Analysis

Open NetSim, Select Examples ▷IOT-WSN ▷Wireless Sensor Networks ▷CAP Time Analysis, then click on the tile in the middle panel to load the example as shown in Figure 4-43.

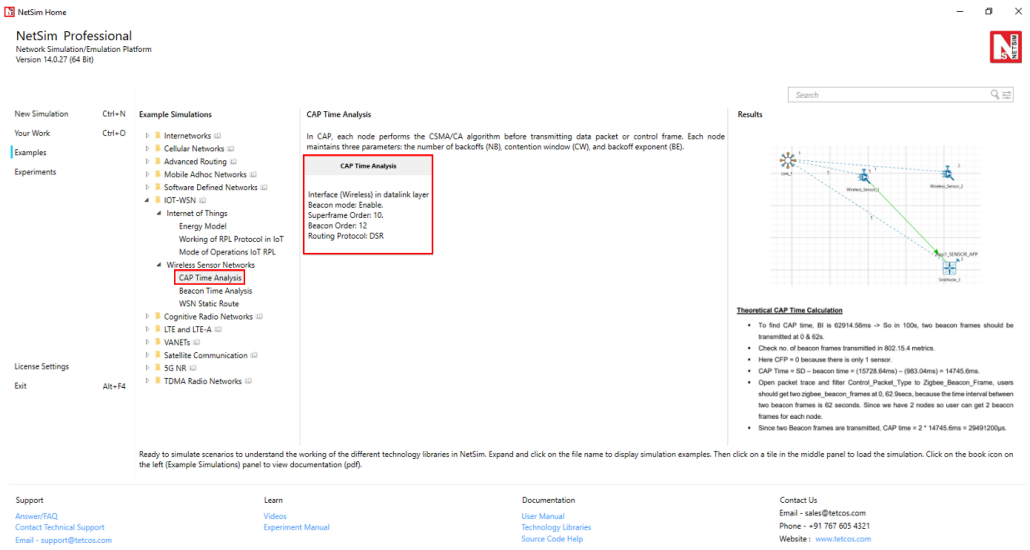


Figure 4-43: *List of scenarios for the example of CAP Time Analysis.*

The following network diagram illustrates what the NetSim UI displays when you open the example configuration file.

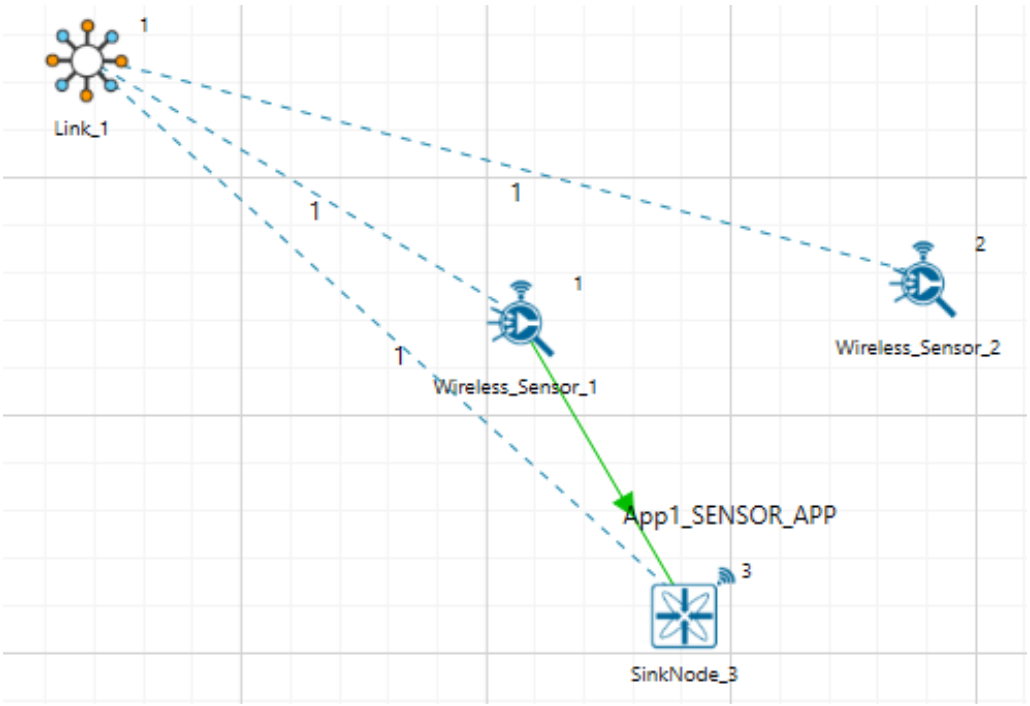


Figure 4-44: *Network set up for studying the CAP Time Analysis.*

Settings done in example config file

- The following grid properties are already set, as Manually Via Click and Drop.
- In the SinkNode, the Super frame mode and Beacon mode are set to 10 and 12, respectively. To configure it, click on the SinkNode. On the right side, expand the property panel, go to the data link

layer of the Interface (Zigbee) layer, and set the Superframe Order (SO) to 10 and the Beacon Order (BO) to 12.

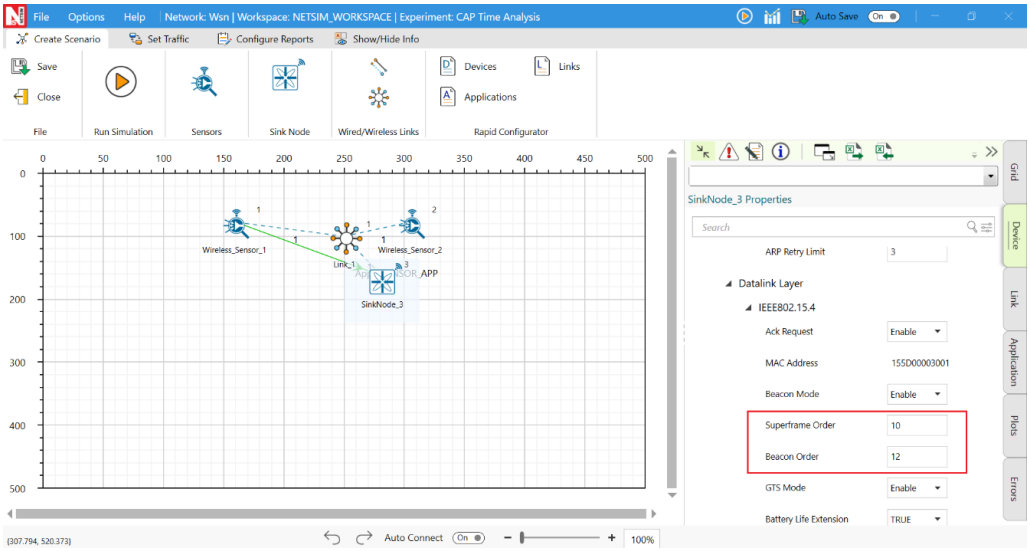


Figure 4-45: *Datalink layer Properties window for Sink node.*

- In the position layer of wireless sensors and the sink, the mobility model is set to Random Waypoint.
- In Adhoc Link Properties change Channel characteristics ▷ Path Loss only, Path Loss Model ▷ Log Distance and path loss exponent ▷ 2.

Create sensor traffic between Wireless Sensor 1 and WSN Sink 3 from the Set Traffic tab in the top ribbon. Click on the created application, set the transport layer protocol to UDP, and leave the other properties as default.

Enable Packet trace from the configure report in the ribbon on the top.

Run the Simulation for 100 sec.

Theoretical CAP Time Calculation

- To find CAP time, BI is 62914.56ms ▷ So in 100s, two beacon frames should be transmitted at 0 & 62s.
- Check no. of beacon frames transmitted in 802.15.4 metrics.
- Here CFP = 0 because there is only 1 sensor.
- $CAP\ Time = SD - beacon\ time = (15728.64ms) - (983.04ms) = 14745.6ms.$
- Open packet trace and filter Control Packet Type to Zigbee Beacon Frame, users should get two ZigBee beacon frames at 0, 62.9secs, because the time interval between two beacon frames is 62 seconds. Since we have 2 nodes, the user can get 2 beacon frames for each node.
- Since two Beacon frames are transmitted,

$$CAP\ time = 2 \times 14745.6ms = 29491200\ \mu s.$$

NetSim Results:

- In the NetSim simulation results window, click on Additional Metrics and scroll down to the IEEE 802.15.4 Metrics table. Compare the theoretical CAP time with the CAP time in the IEEE 802.15.4 Metrics table.

- CAP time = 29491200.0000 Microsec.

IEEE802.15.4_Metrics

IEEE802.15.4_Metrics

Device ID	PacketTransmitted	PacketReceived	CCAAttempt	SuccessfulCCA	BeaconTransm	BeaconTime(Micro secc	CAPTime(Micro sec	CFPTime(Micrc
1	93	12	378	374	0	0.0000	0.0000	0.0000
2	5	15	24	24	0	0.0000	0.0000	0.0000
3	11	92	54	54	2	1966080.0000	29491200.0000	0.0000

Figure 4-46: *IEEE802.15.4 Metrics Table in Results Window.*

4.2.3 Static Routing in WSN

Open NetSim, Select Examples ▹IOT-WSN ▹Wireless Sensor Networks ▹WSN Static Route then click on the tile in the middle panel to load the example as shown in Figure 4-47.

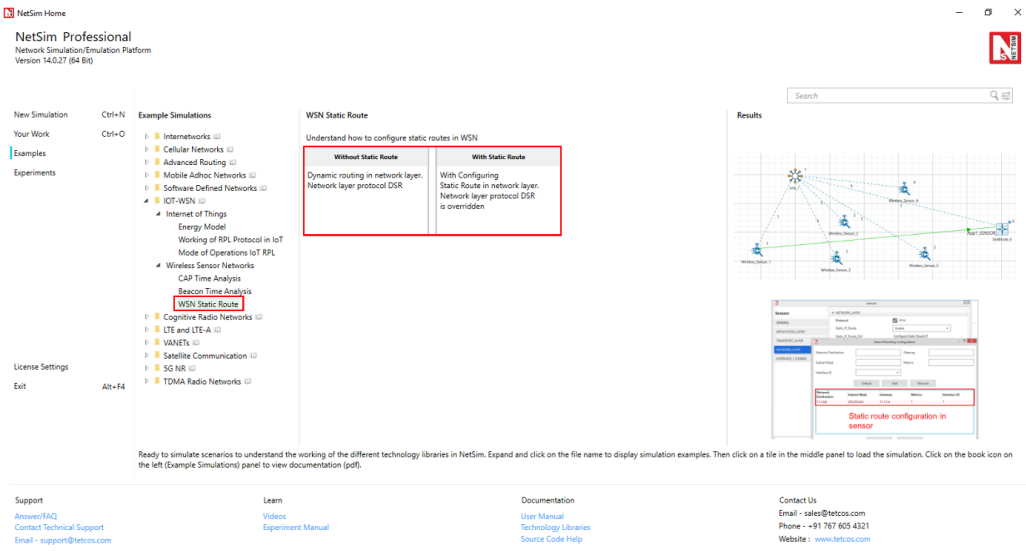


Figure 4-47: *List of scenarios for the example of WSN Static Route.*

The following network diagram illustrates what the NetSim UI displays when you open the example configuration file.

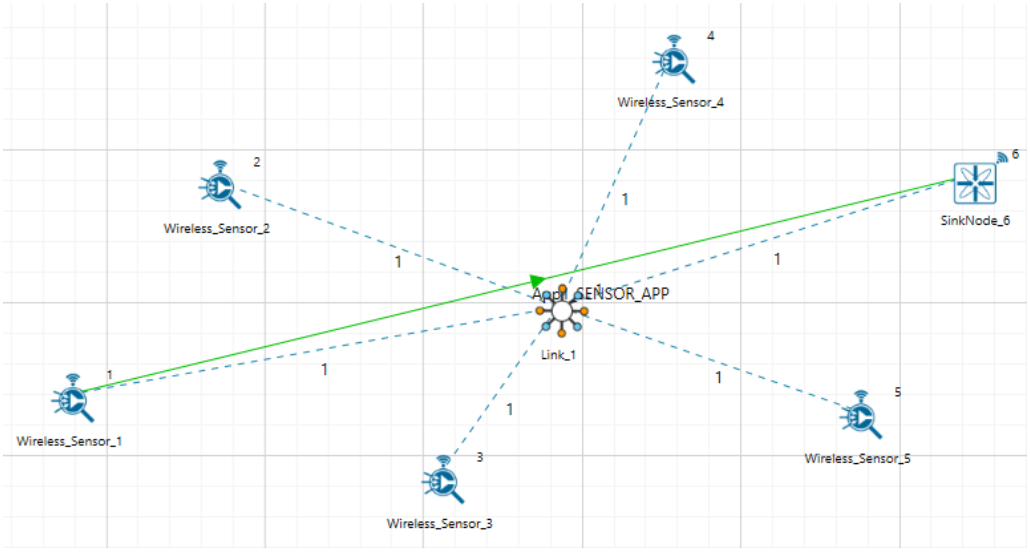
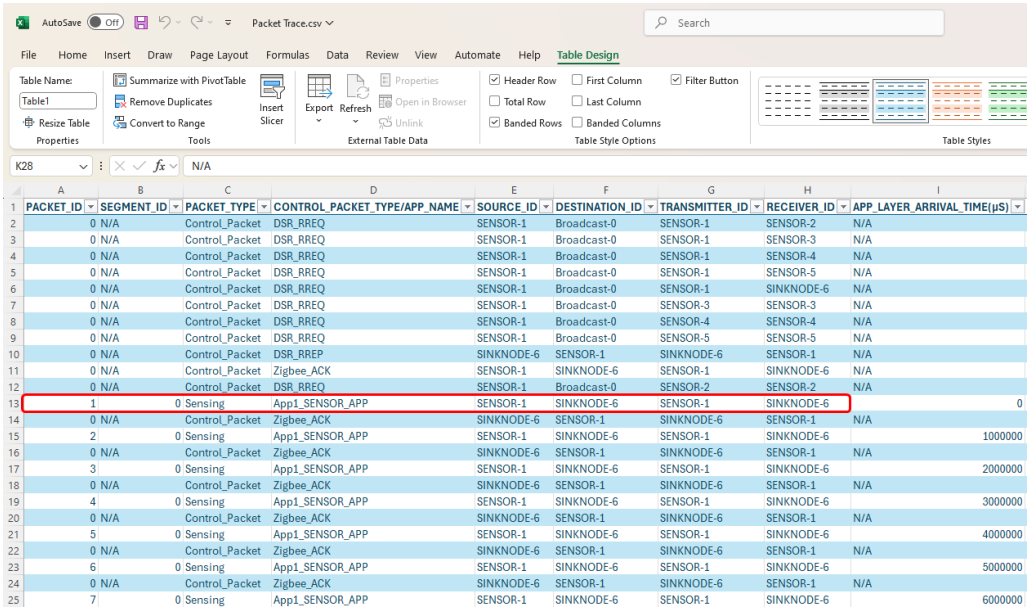


Figure 4-48: *Network set up for studying the WSN Static Route.*

Case 01: Without Static Route

Settings done in the network

- The following grid properties are already set, as Manually via Click and Drop.
- The sensor application is created between Sensor 1 and 6 from the Set Traffic tab in the top ribbon.
- Packet trace is enabled from the configure reports tab.
- Simulation time is set to 100 sec.
- Results: Open the packet trace and observe the data flow. Sensor 1 sends packets directly to Sinknode 6.



	PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER_ARRIVAL_TIME(µS)
2	0	N/A	Control_Packet	DSR_RREQ	SENSOR-1	Broadcast-0	SENSOR-1	SENSOR-2	N/A
3	0	N/A	Control_Packet	DSR_RREQ	SENSOR-1	Broadcast-0	SENSOR-1	SENSOR-3	N/A
4	0	N/A	Control_Packet	DSR_RREQ	SENSOR-1	Broadcast-0	SENSOR-1	SENSOR-4	N/A
5	0	N/A	Control_Packet	DSR_RREQ	SENSOR-1	Broadcast-0	SENSOR-1	SENSOR-5	N/A
6	0	N/A	Control_Packet	DSR_RREQ	SENSOR-1	Broadcast-0	SENSOR-1	SINKNODE-6	N/A
7	0	N/A	Control_Packet	DSR_RREQ	SENSOR-1	Broadcast-0	SENSOR-3	SENSOR-3	N/A
8	0	N/A	Control_Packet	DSR_RREQ	SENSOR-1	Broadcast-0	SENSOR-4	SENSOR-4	N/A
9	0	N/A	Control_Packet	DSR_RREQ	SENSOR-1	Broadcast-0	SENSOR-5	SENSOR-5	N/A
10	0	N/A	Control_Packet	DSR_RREP	SINKNODE-6	SENSOR-1	SINKNODE-6	SENSOR-1	N/A
11	0	N/A	Control_Packet	Zigbee_ACK	SENSOR-1	SINKNODE-6	SENSOR-1	SINKNODE-6	N/A
12	0	N/A	Control_Packet	DSR_RREQ	SENSOR-1	Broadcast-0	SENSOR-2	SENSOR-2	N/A
13	1	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-1	SINKNODE-6	0
14	0	N/A	Control_Packet	Zigbee_ACK	SINKNODE-6	SENSOR-1	SINKNODE-6	SENSOR-1	N/A
15	2	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-1	SINKNODE-6	1000000
16	0	N/A	Control_Packet	Zigbee_ACK	SINKNODE-6	SENSOR-1	SINKNODE-6	SENSOR-1	N/A
17	3	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-1	SINKNODE-6	2000000
18	0	N/A	Control_Packet	Zigbee_ACK	SINKNODE-6	SENSOR-1	SINKNODE-6	SENSOR-1	N/A
19	4	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-1	SINKNODE-6	3000000
20	0	N/A	Control_Packet	Zigbee_ACK	SINKNODE-6	SENSOR-1	SINKNODE-6	SENSOR-1	N/A
21	5	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-1	SINKNODE-6	4000000
22	0	N/A	Control_Packet	Zigbee_ACK	SINKNODE-6	SENSOR-1	SINKNODE-6	SENSOR-1	N/A
23	6	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-1	SINKNODE-6	5000000
24	0	N/A	Control_Packet	Zigbee_ACK	SINKNODE-6	SENSOR-1	SINKNODE-6	SENSOR-1	N/A
25	7	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-1	SINKNODE-6	6000000

Figure 4-49: *NetSim packet trace showing packet flow for case 1: Without static routing.*

Case 02: With Static Route

Settings done in the network.

- For the above sample (without static route), we have configured the static routes for sensors. The following steps explain how to configure static routes.

Configuring Static Routes

Click on the wireless sensor 1. In the right property panel, go to the network layer under IPv4 and enable Static IP Route.

Then, click on via GUI in Configure Static IP Route and set the Network Destination, Gateway, Subnet Mask, Metrics, and Interface ID as shown in the screenshot below. Click add and then OK.

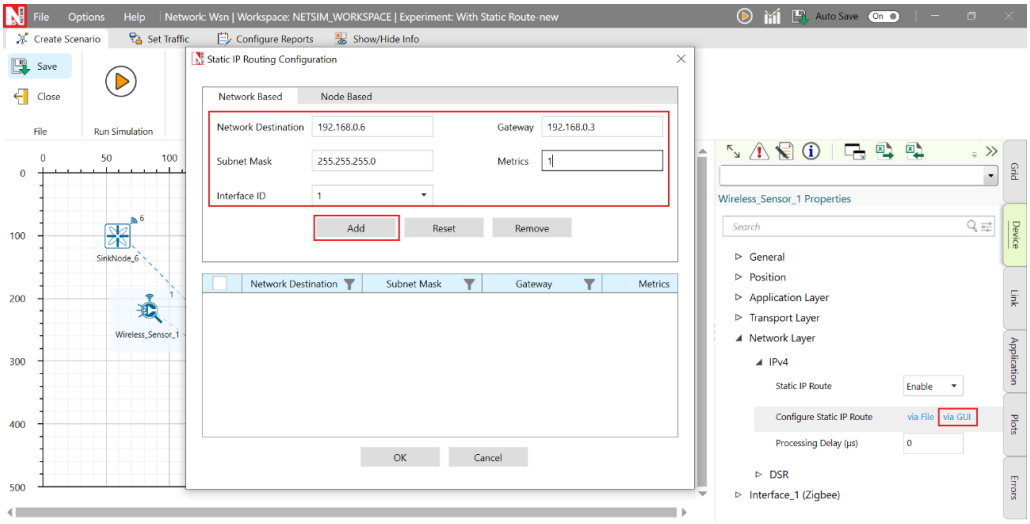


Figure 4-50: *Static IP Routing Configuring window.*

Similarly, configure static routes for other sensors as follows:

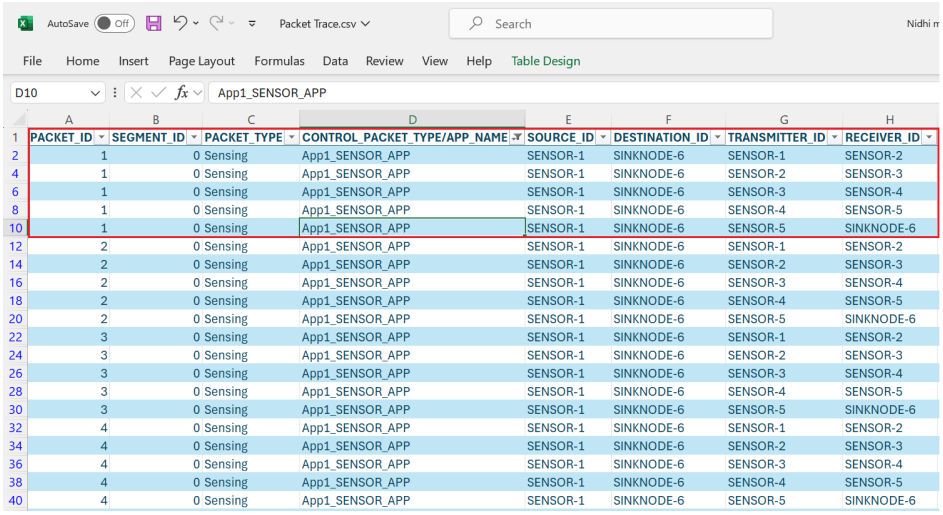
Table 4-9: *Static Route Configuration for Sensors.*

Device	Network Destination	Gateway	Subnet Mask	Metrics	Interface ID
Wireless Sensor 1	192.168.0.7	192.168.0.3	255.255.255.0	1	1
Wireless Sensor 2	192.168.0.7	192.168.0.4	255.255.255.0	1	1
Wireless Sensor 3	192.168.0.7	192.168.0.5	255.255.255.0	1	1
Wireless Sensor 4	192.168.0.7	192.168.0.6	255.255.255.0	1	1
Wireless Sensor 5	192.168.0.7	192.168.0.7	255.255.255.0	1	1

- After setting the properties click on run simulation for 100 sec.

Results

- Open the packet trace and observe the packet flow. The data will be transmitted according to the statically configured routes.
- SENSOR 1 → SENSOR 2 → SENSOR 3 → SENSOR 4 → SENSOR 5 → SINK NODE



	A	B	C	D	E	F	G	H
1	PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
2	1	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-1	SENSOR-2
4	1	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-2	SENSOR-3
6	1	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-3	SENSOR-4
8	1	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-4	SENSOR-5
10	1	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-5	SINKNODE-6
12	2	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-1	SENSOR-2
14	2	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-2	SENSOR-3
16	2	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-3	SENSOR-4
18	2	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-4	SENSOR-5
20	2	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-5	SINKNODE-6
22	3	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-1	SENSOR-2
24	3	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-2	SENSOR-3
26	3	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-3	SENSOR-4
28	3	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-4	SENSOR-5
30	3	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-5	SINKNODE-6
32	4	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-1	SENSOR-2
34	4	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-2	SENSOR-3
36	4	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-3	SENSOR-4
38	4	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-4	SENSOR-5
40	4	0	Sensing	App1_SENSOR_APP	SENSOR-1	SINKNODE-6	SENSOR-5	SINKNODE-6

Figure 4-51: *NetSim packet trace showing packet flow for case 2: With static routing.*

4.2.4 Analysing throughput, latency and energy consumption as we scale the number of sensors

We examine a scenario in which sensors are arranged in a circular configuration around a central sink, with all sensors positioned within the carrier sense range of each other. As the number of sensors increases, we analyse throughput, latency, and energy consumption in ZigBee's (802.15.4) beaconless and beacon enabled modes across different packet sizes.

Network Layout

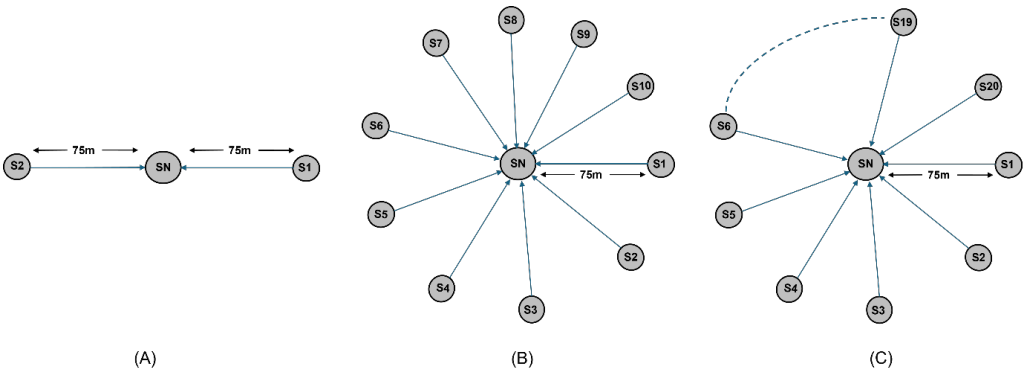


Figure 4-52: *Schematic of network layout. In all cases the sensors are 75m from the sink. (A) Scenario with 2 sensors and sink node (SN), (B) Scenario with 10 sensors and a sink, (C) Scenario with 20 sensors and a sink node. The sources of traffic are Sensor 1 (S1), Sensor 2 (S2), Sensor 3 (S3) etc. and the destination is the Sink Node (SN). The transmission range of the sensors is 177m, while the carrier sense range is 223m.*

Network Scenario Open NetSim, Select Examples ▷ IOT-WSN ▷ Wireless Sensor Networks ▷ Analyzing throughput, latency and energy consumption as we scale the number of sensors ▷ Analyzing Throughput, Latency, and Energy Consumption in Beacon-Disabled Mode and then click on the tile in the middle panel to load the example as shown in Figure 4-53.

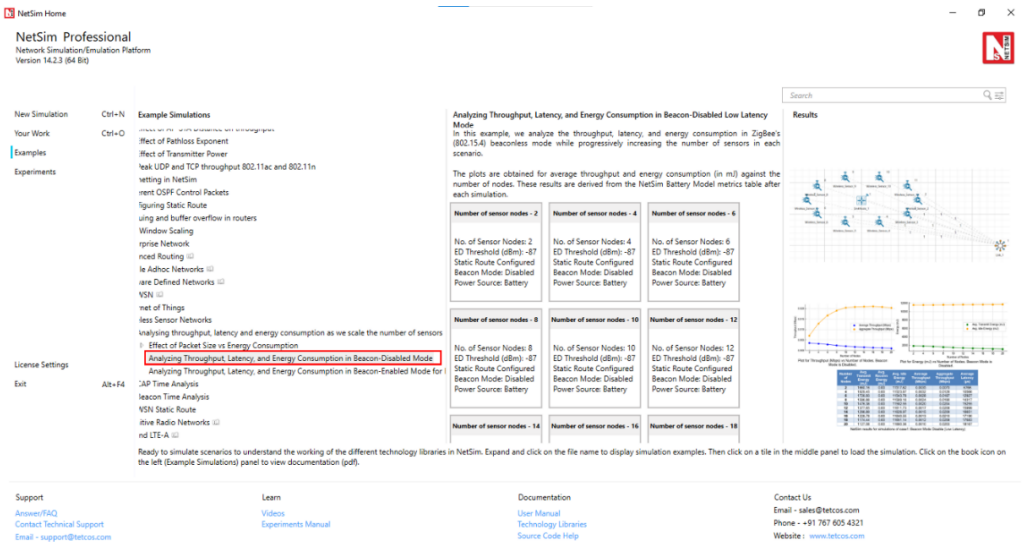


Figure 4-53: *List of scenarios for the example of Analyzing Throughput, Latency, and Energy Consumption in Beacon-Disabled Mode.*

The following network diagram illustrates what the NetSim UI displays when you open the example configuration file.

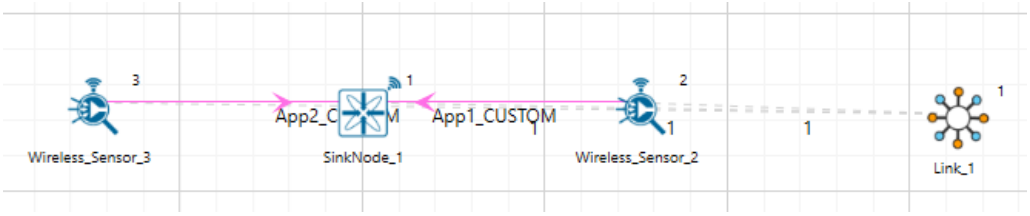


Figure 4-54: *Network scenario modelled in NetSim. This scenario consists of 2 sensors and 1 sink node where the distance between every sensor and the sink node is 75m.*

Case 1:

- Create a scenario with 2 Sensor nodes and 1 Sink node.
- Distance between the sensors and sink nodes should be 75m.
- Click on Sink node and expand the property panel on right side and disable the beacon mode in datalink layer of Interface (Zigbee) properties.

Table 4-10: *Device properties for sensors.*

Device Properties	
Network layer properties	
Routing protocol	DSR
Interface (Wireless) properties	
Physical layer properties	
ED Threshold (dBm)	−87
Power	Battery

- Similarly repeat this step for all other sensors.
- Configure the Static Route from Sensors to Sink Node as shown below.
- Static routes were configured in each source node such that the data gets transmitted directly from source to destination without any dynamic route formation by the routing protocols.
- Static routes (whereby N_i always transmits to $N_{(i+1)}$) are set to ensure single hop transmission. Thereby each node transmits data to the next-hop node according to the topology.
- To set the static routes, go to Wireless Sensor properties ▷ Network Layer ▷ Enable Static Route IP and click on Configure static route via GUI and update as below.

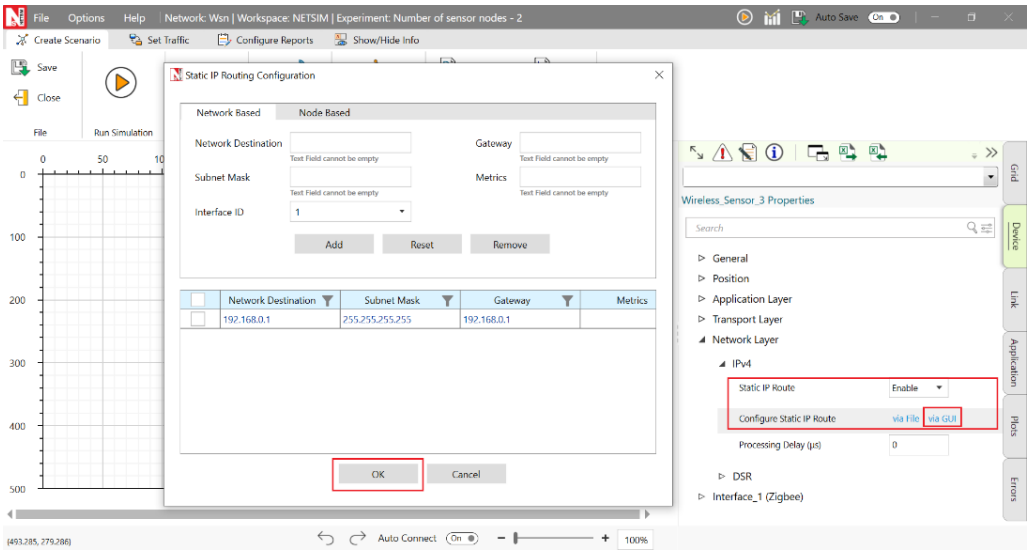


Figure 4-55: *Static route configuration for sensor 1.*

- The Static route IPs were configured in Wireless Sensor 1, Wireless Sensor 2, Wireless Sensor 3 etc. and Wireless Sensor 20 as shown below:

Table 4-11: *Static route configured in devices.*

Device	Network Destination	Gateway	Subnet Mask	Metrics	Interface ID
Wireless Sensor 1	192.168.0.1	192.168.0.1	255.255.255.255	1	1
Wireless Sensor 2	192.168.0.1	192.168.0.1	255.255.255.255	1	1
Wireless Sensor 3	192.168.0.1	192.168.0.1	255.255.255.255	1	1
Wireless Sensor 4	192.168.0.1	192.168.0.1	255.255.255.255	1	1
Wireless Sensor i	192.168.0.1	192.168.0.1	255.255.255.255	1	1
Wireless Sensor 20	192.168.0.1	192.168.0.1	255.255.255.255	1	1

- Create a CUSTOM application from Sensor to Sink Node by clicking on set traffic tab from the ribbon on the top. Click on created application and expand application property on right panel, set packet size to 50 bytes and Inter arrival time to 100000 μs.

- Run the Simulation for 1000 seconds.
- Similarly increase the number of nodes to 4, 6, 8, 10, 12, 14, 16, 18, 20 and record the results.

Results Case 1: Beacon Mode Disabled (Low Latency)

Results for throughput and latency can be observed from the application metrics window of simulation results window and to observe transmit energy, receive energy and idle energy, click on additional metrics in the left and scroll down to the battery model metrics.

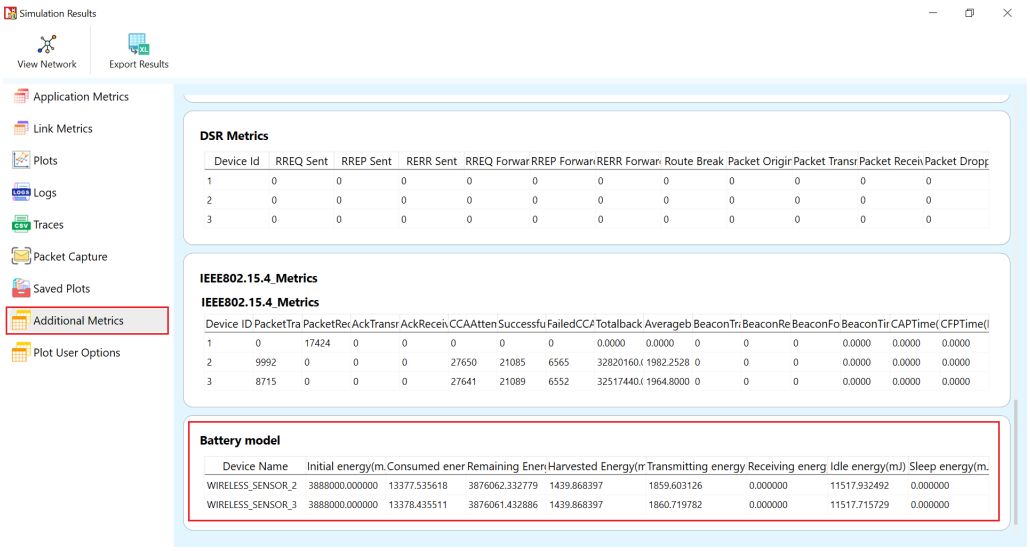


Figure 4-56: *Observe battery model metrics from NetSim simulation results window.*

Table 4-12: *NetSim results for simulations of case 1.*

Number of sensor nodes	Avg. Transmit Energy (mJ)	Avg. Receive Energy (mJ)	Avg. Idle Energy (mJ)	Average Throughput (Mbps)	Aggregate Throughput (Mbps)	Average Latency (Delay) (μs)
2	1860.16	0.00	11517.82	0.0035	0.0070	6768
4	1829.45	0.00	11523.87	0.0032	0.0128	10566
6	1726.93	0.00	11543.78	0.0028	0.0167	12927
8	1596.88	0.00	11569.16	0.0024	0.0190	14317
10	1476.38	0.00	11592.55	0.0020	0.0204	15255
12	1377.65	0.00	11611.73	0.0017	0.0208	15999
14	1298.89	0.00	11626.97	0.0015	0.0209	16651
16	1228.78	0.00	11640.55	0.0013	0.0210	17190
18	1174.44	0.00	11651.14	0.0012	0.0208	17683
20	1127.08	0.00	11660.36	0.0010	0.0203	18167

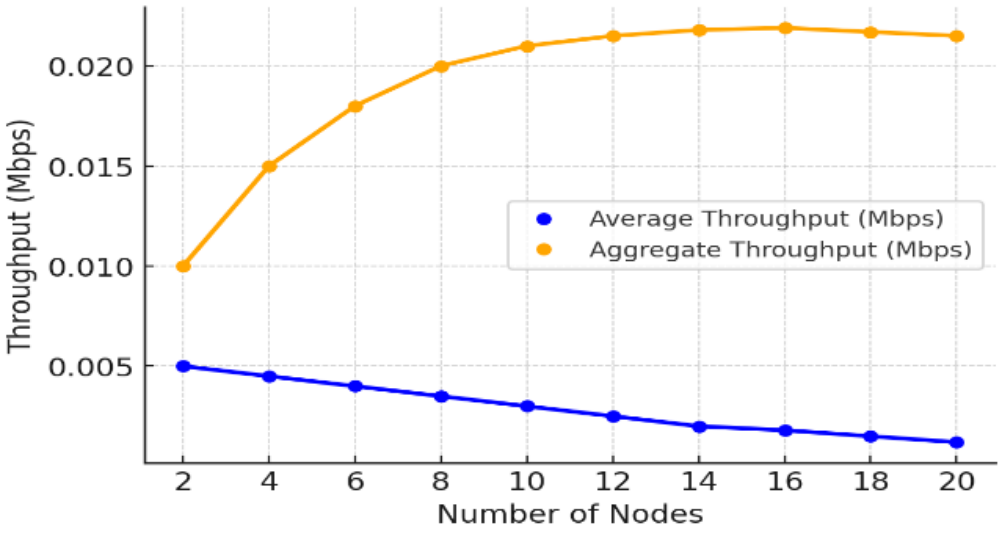


Figure 4-57: *Plot for Throughput (Mbps) vs Number of Nodes. Beacon Mode is Disabled.*

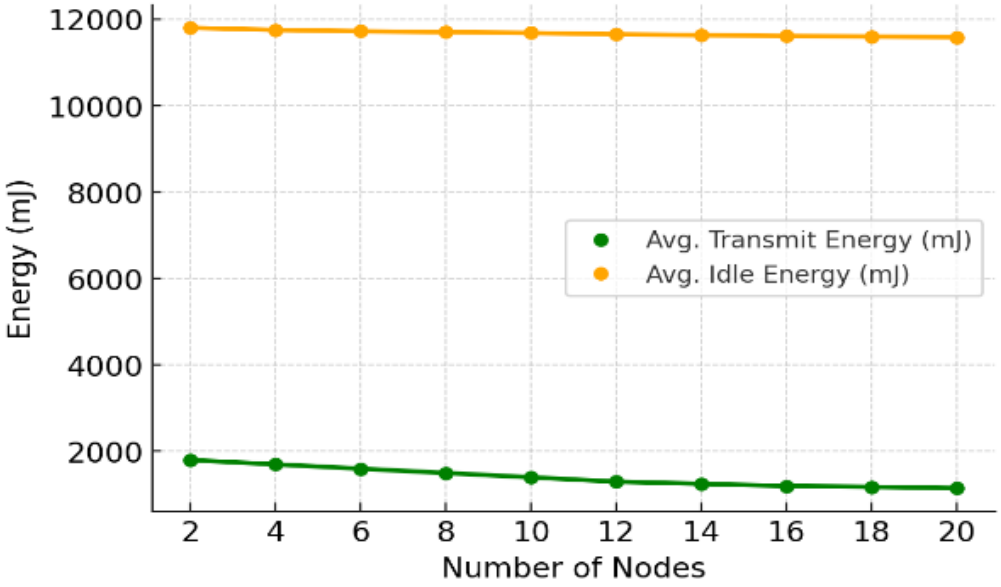


Figure 4-58: *Plot for Energy (mJ) vs Number of Nodes. Beacon Mode is Disabled.*

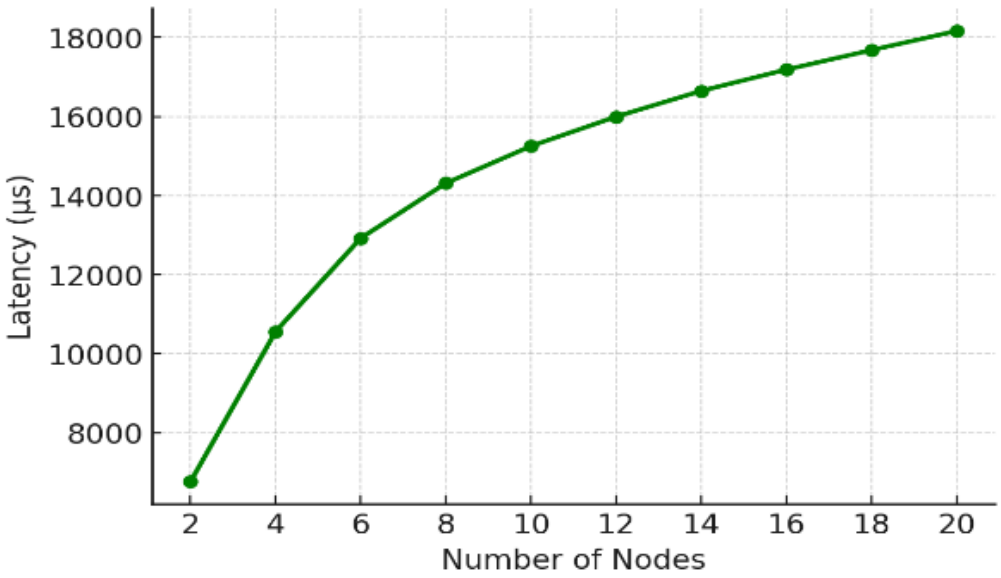


Figure 4-59: *Plot for Latency (µs) vs Number of Nodes. Beacon Mode is Disabled.*

Discussion The aggregate throughput increases initially but then decreases by increasing the number of nodes. The initial increase is because for small number-of-nodes the contention is less and increasing the node count increases the channel utilization.

With beacon mode disabled, all nodes remain active throughout the simulation, constantly ready to receive or transmit data. The transmitting energy per node decreases as more nodes are added to the network, which is due to the distribution of transmissions among more sensors. However, this operational mode results in a high average idle energy as nodes consume power even when they are not actively transmitting data.

This transmission mode is suitable for applications that prioritize low latency over energy efficiency.

Network Scenario Open NetSim, Select Examples ▷ IOT-WSN ▷Wireless Sensor Networks ▷ Analyzing throughput, latency and energy consumption as we scale the number of sensors ▷ Analyzing Throughput, Latency, and Energy Consumption in Beacon-Enabled Mode for Energy Efficiency then click on the tile in the middle panel to load the example as shown in Figure 4-60.

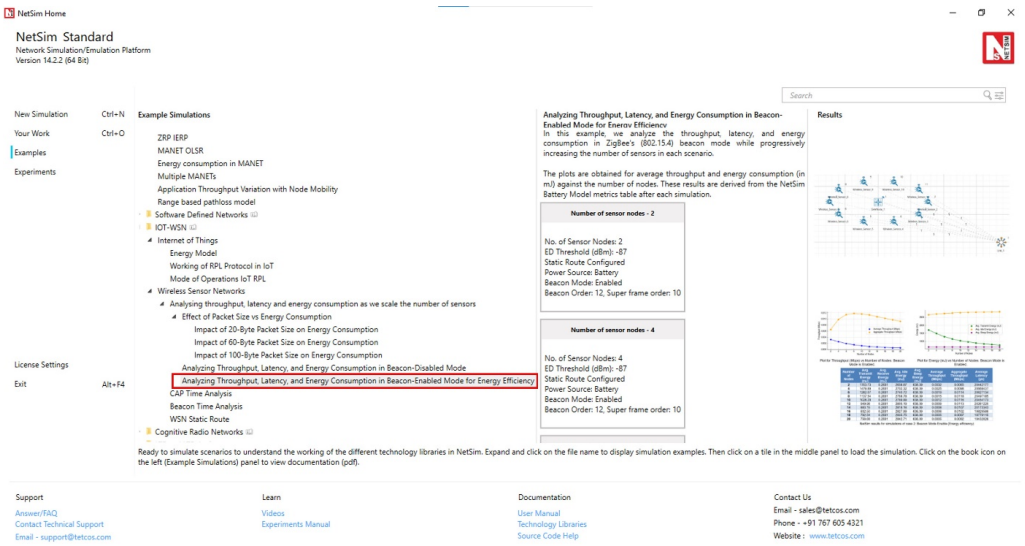


Figure 4-60: *List of scenarios for the example of Analyzing Throughput, Latency, and Energy Consumption in Beacon-Enabled Mode for Energy Efficiency.*

Case 2:

- Consider the same scenario as Case 1
- Go to Sink Node ▷ Interface ZIGBEE ▷ Datalink Layer, Set the Beacon Mode to Enable and Set the Beacon Order to 12 and Super frame Order 10.
- Run the Simulation for 1000 seconds.

Results Case 2: Beacon Mode Enabled (Energy efficiency)

Table 4-13: *NetSim results for simulations of case 2.*

No. of nodes	Avg. Tx Energy (mJ)	Avg. Rx Energy (mJ)	Avg. Idle Energy (mJ)	Avg. Sleep Energy (mJ)	Avg. Throughput (Mbps)	Agg. Throughput (Mbps)	Avg. Latency (μs)
2	1703.73	0.2831	2658.87	638.39	0.0032	0.0065	20442177
4	1479.89	0.2831	2702.32	638.39	0.0025	0.0098	20668437
6	1282.07	0.2831	2740.72	638.39	0.0019	0.0114	20627134
8	1137.54	0.2831	2768.78	638.39	0.0015	0.0118	20497185
10	1028.29	0.2831	2789.99	638.39	0.0012	0.0116	20454173
12	949.96	0.2831	2805.19	638.39	0.0009	0.0113	20261225
14	883.15	0.2831	2818.16	638.39	0.0008	0.0107	20173343
16	832.50	0.2831	2827.99	638.39	0.0006	0.0102	19829588
18	792.54	0.2831	2835.75	638.39	0.0005	0.0097	19779115
20	756.68	0.2831	2842.71	638.39	0.0005	0.0092	19432628

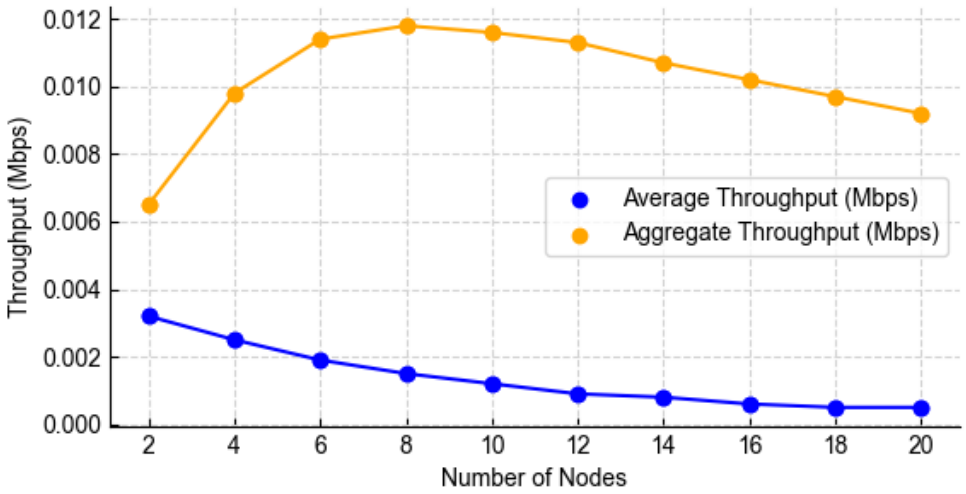


Figure 4-61: *Plot for Throughput (Mbps) vs Number of Nodes. Beacon Mode is Enabled.*

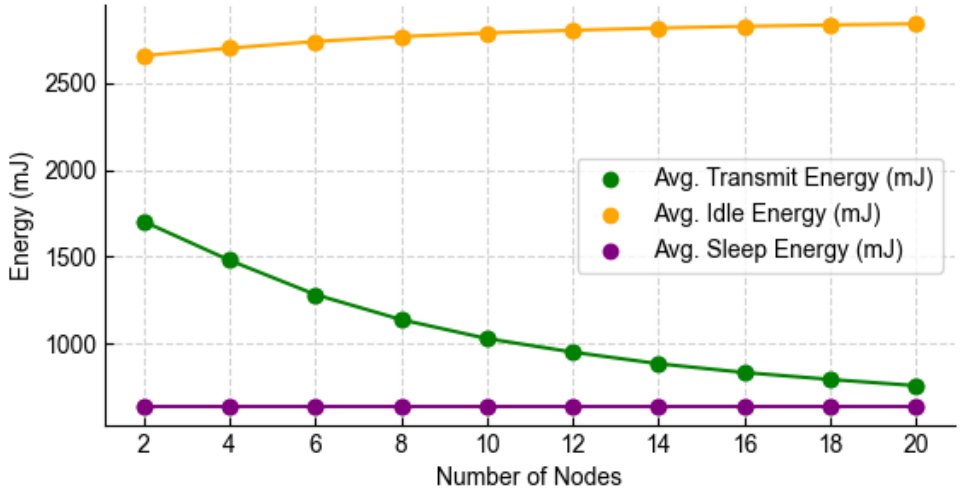


Figure 4-62: *Plot for Energy (mJ) vs Number of Nodes. Beacon Mode is Enabled.*

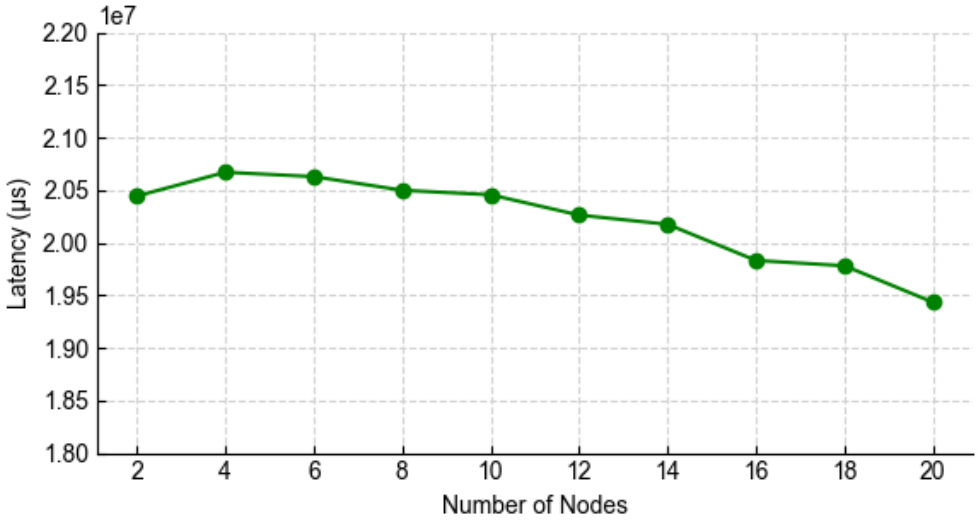


Figure 4-63: *Latency (µs) vs Number of Nodes. Beacon Mode is Enabled.*

Discussion Case 2 introduces beacon-enabled mode in the network, significantly altering energy consumption patterns and network performance. When beacon mode is enabled, nodes in the network operate with defined periods for active and sleep states. Nodes wake up at specific beacon intervals to transmit and receive data, and then go back to sleep. This operation significantly reduces the average idle energy consumption because nodes are not constantly powered on waiting to receive or transmit data. Instead, they spend a substantial portion of time in the low-power sleep state, which is reflected in the sleep energy metrics in the results.

However, this also introduces a substantial increase in average latency, as nodes must wait for their turn to transmit during the beacon intervals, leading to delays in packet forwarding. This is reflected in the latency values, which are orders of magnitude higher than in Case 1. Throughput slightly decreases compared to Case 1, which is due to the time spent in low-power states, reducing the effective transmission time.

This transmission mode is suitable for applications that prioritize energy efficiency over latency.

Network scenario Open NetSim, Select Examples ▷ IOT-WSN ▷ Wireless Sensor Networks ▷ Analyzing throughput, latency and energy consumption as we scale the number of sensors ▷ Effect of Packet Size vs Energy Consumption, then click on the tile in the middle panel to load the example as shown in Figure 4-64.

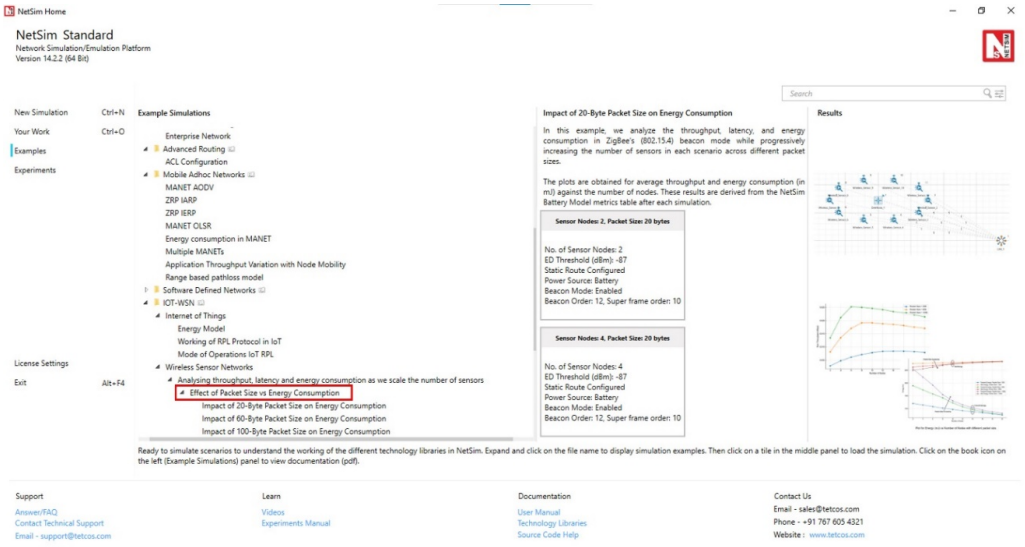


Figure 4-64: *List of scenarios for the example of Effect of Packet Size vs Energy Consumption.*

Case 3: Effect of Packet Size vs Energy Consumption

- Consider the same scenario as Case-2
- Create a CUSTOM Application with Source ID as Sensor and Destination ID as Sink Node, with the Packet Size 20 Bytes and Inter Arrival Time 100000 μs.
- Run the Simulation for 1000 seconds.
- Record the results from results dashboard.
- Similarly vary the packet size in application to 60B and 100B respectively.
- Rerun the simulation for 1000 seconds. Record the results from the results dashboard.

Results

Table 4-14: *Results table of Case-3.*

No. of nodes	Pkt Size (B)	Avg. Tx Energy (mJ)	Avg. Rx Energy (mJ)	Avg. Idle Energy (mJ)	Avg. Sleep Energy (mJ)	Avg. Throughput (Mbps)	Agg. Throughput (Mbps)	Avg. Latency (μs)
2	20	1200.53	0.28	2756.56	638.4	0.0014	0.0028	20319671
4	20	1142.49	0.28	2767.83	638.4	0.0012	0.0047	21071912
6	20	1081.05	0.28	2779.75	638.4	0.001	0.0061	21440249
8	20	1023.47	0.28	2790.93	638.4	0.0009	0.0071	21903767
10	20	969.01	0.28	2801.5	638.4	0.0008	0.0078	22150841
12	20	919.89	0.28	2811.04	638.4	0.0007	0.0083	22737334
14	20	879.28	0.28	2818.92	638.4	0.0006	0.0084	22943219
16	20	839.94	0.28	2826.55	638.4	0.0005	0.0084	23845867
18	20	790.46	0.28	2836.16	638.4	0.0005	0.0082	33464665
20	20	746.35	0.28	2844.72	638.4	0.0004	0.0079	44959110
2	60	1898.03	0.28	2621.16	638.4	0.004	0.0081	20736084
4	60	1731.53	0.28	2653.48	638.4	0.0034	0.0135	21670506
6	60	1569.88	0.28	2684.86	638.4	0.0028	0.017	22822081
8	60	1424.32	0.28	2713.12	638.4	0.0024	0.0191	23630100
10	60	1223.77	0.28	2752.05	638.4	0.0019	0.0191	53919154
12	60	1074.47	0.28	2781.03	638.4	0.0016	0.0188	79323930
14	60	967.4	0.28	2801.81	638.4	0.0013	0.0185	95181466
16	60	885.83	0.28	2817.65	638.4	0.0011	0.0181	107682323
18	60	821.14	0.28	2830.21	638.4	0.001	0.0175	115764304
20	60	770.42	0.28	2840.05	638.4	0.0009	0.017	123905731
2	100	2556.55	0.28	2493.33	638.4	0.0067	0.0134	21258705
4	100	2231.26	0.28	2556.48	638.4	0.0053	0.0211	22658685
6	100	1919.71	0.28	2616.95	638.4	0.0042	0.0252	27512241
8	100	1545.9	0.28	2689.52	638.4	0.0031	0.025	73068592
10	100	1310.88	0.28	2735.14	638.4	0.0025	0.0246	100074374
12	100	1151.74	0.28	2766.03	638.4	0.002	0.0241	118197099
14	100	1035.23	0.28	2788.65	638.4	0.0017	0.0234	129367238
16	100	944.31	0.28	2806.3	638.4	0.0014	0.0229	138470037
18	100	876.48	0.28	2819.46	638.4	0.0012	0.0221	145188141
20	100	819.23	0.28	2830.58	638.4	0.0011	0.0214	150341336

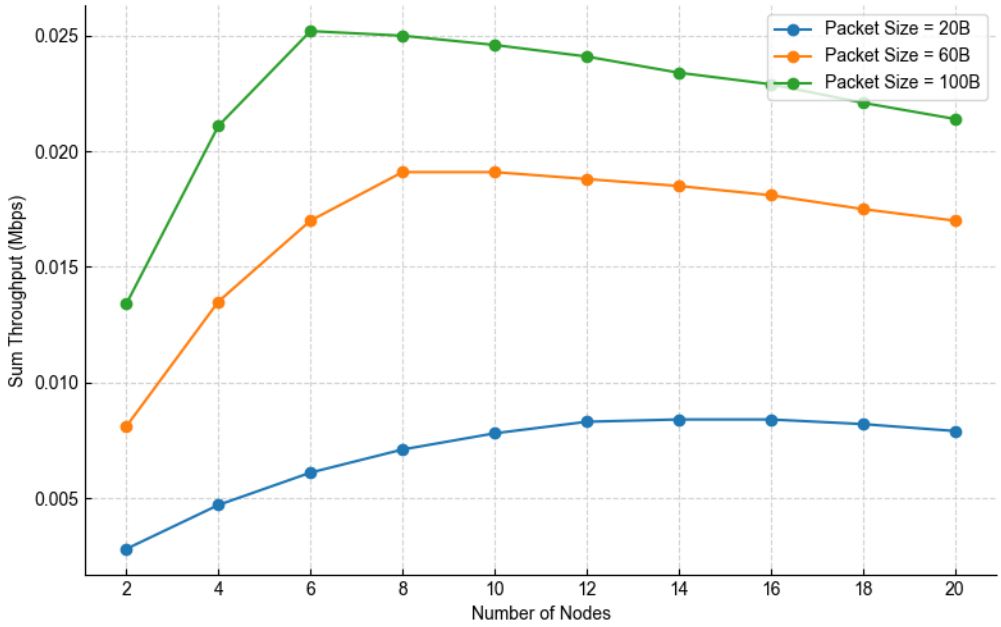


Figure 4-65: *Plot for Sum Throughput (Mbps) vs Number of Nodes with different packet size.*

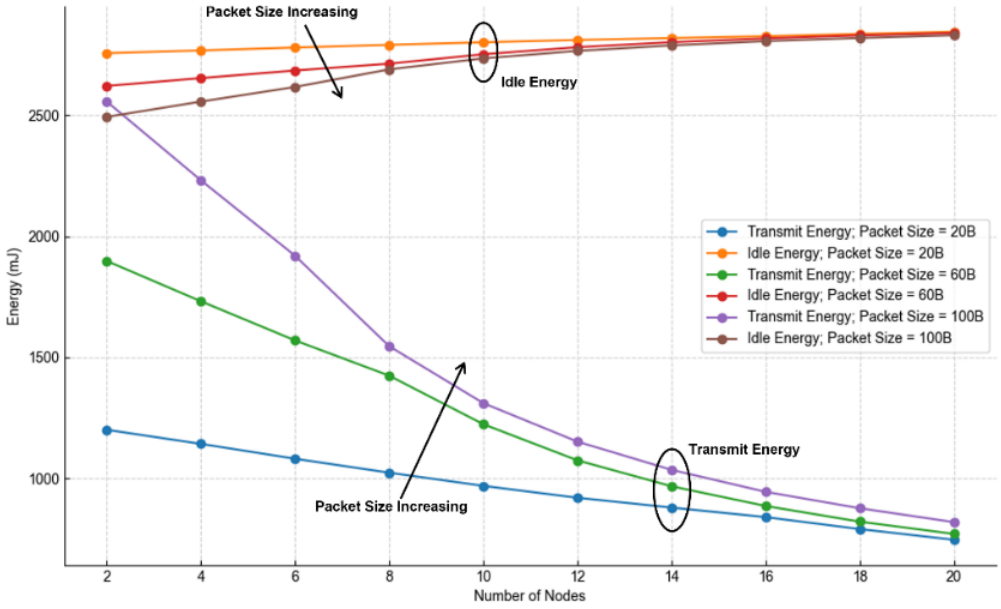


Figure 4-66: *Plot for Energy (mJ) vs Number of Nodes with different packet size.*

Discussion Table 4-14 presents detailed metrics for energy consumption, throughput, and latency across various numbers of nodes and packet sizes (20B, 60B, and 100B).

- Throughput Observations:
  - Average Throughput: This metric decreases with the increase in the number of nodes for all packet sizes. As more nodes are added, the data transmission per node reduces, due to increased contention and collisions on the network.
  - Aggregate Throughput: In contrast to average throughput, aggregate throughput increases initially as the number of nodes increases but then starts to decrease as further nodes are added. The initial increase is because for small number-of-nodes the contention is less and increasing the node count

increases the channel utilization. However, beyond a threshold with more nodes, the likelihood of collisions increases due to more frequent attempts to access the channel simultaneously.

- Energy Consumption Observations:
  - Transmit Energy: Average transmit energy decreases as the number of nodes increases. This is due to the distribution of transmissions among more nodes.
  - Receive and Sleep energy: Receive energy and sleep energy remain relatively consistent across different numbers of nodes and packet sizes. This indicates that the receive and sleep mode energy consumption is not dependent on the network size or the packet size but is dependent on the node's configuration (hardware) settings.
- Insights:
  - Packet Size Efficiency: Larger packet sizes lead to more effective use of the network capacity, as observed by increase in aggregate throughput with packet size. Larger packets can better amortize the overheads of transmission, leading to improved application throughputs.
  - Energy vs. Throughput Trade-offs: There is an observable trade-off between energy consumption and throughput. While larger packets improve throughput, they also require more transmit energy. Conversely, while smaller packets lead to lower transmit energy, they achieve lower throughput.

5 IOT-WSN Experiments in NetSim

Apart from examples, in-built experiments are also available in NetSim. Examples help the user understand the working of features in NetSim. Experiments are designed to help the user (usually students) learn networking concepts through simulation. The experiments contain objective, theory, set-up, results, and inference. The following experiments are available in the Experiments manual (pdf file).

1. One Hop IoT Network over IEEE 802.15.4
2. IoT – Multi-Hop Sensor-Sink Path
3. Performance Evaluation of a Star Topology IoT Network
4. Study the 802.15.4 Superframe Structure and analyze the effect of Superframe order on throughput.

6 Reference Documents

1. O. Landsiedel, K. Wehrle and S. Gotz, "Accurate Prediction of Power Consumption in Sensor Networks. University of Tübingen, Germany," 2005.
2. T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. Vasseur and R. Alexander, "IETF RFC 6550: RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks".
3. "IEEE Standard 802.15.4-2011: Low-Rate Wireless Personal Area Networks (LR-WPANs)".

7 Latest FAQs

Up-to-date FAQs on NetSim's IoT/WSN library are available at https://tetcos.freshdesk.com/support/solutions/folders/14000105117