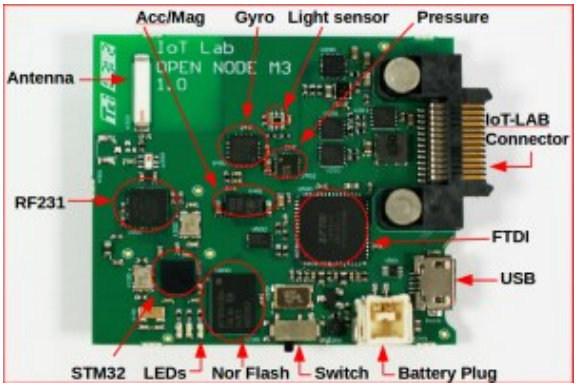
Premises

IoT-LAB M3

The STM32F401RE MCU has been replaced by the ARM Cortex M3 STM32F103REY MCU.

The M3 open node is based on a STM32 (ARM Cortex M3) micro-controller. Like the WSN node this next generation contains a set of sensors and a radio interface. Main evolutions are a more powerful 32-bits processing, a new ATMEL radio interface in 2.4 Hz and more sensors.

MCU	ARM Cortex M3, 32-bits, 72 Mhz, 64kB RAM – STM
sensors	• Ambient sensor light – ISL29020 • Atmospheric pressure and temperature – LPS • Tri-axis accelerometer/magnetometer – L3G42 • Tri-axis gyrometer – LSM303DLHC
radio communication	802.15.4 PHY standard, 2.4 Ghz – AT86RF231
power	3,7V LiPo battery, 650 mAh – GMB 063040



https://www.iot-lab.info/docs/boards/iot-lab-m3/

In order for an embedded board to communicate in IPv6 with a host on the Internet, it needs an IPv6 **global** unicast address. To do this, another embedded board play the role of Border Router (BR) and must be added in the network. It will be responsible for propagating an IPv6 global prefix and assign an address to the device. We speak here of BR, because it's on the border between a

radio network (e.g. 802.15.4) and a classic Ethernet network. Technically, the IPv6 traffic between the SSH frontend and the BR radio interface is routed via a virtual network interface (i.e. TUN or TAP interface) and the IPv6 traffic is encapsulated on the BR's serial link.

Selecting an already used prefix may bring to an *"overlaps with routes"* error while creating the IPv6 over serial interface. To see currently used IPv6 prefixes on a site, use this command from its SSH frontend:

```
<login>@grenoble:~$ ip -6 route
```

- with RIOT using ETHOS (Ethernet Over Serial);

During an experiment, boards running an embedded Linux have a public IPv6 address assigned to their Ethernet interface. They are directly accessible:

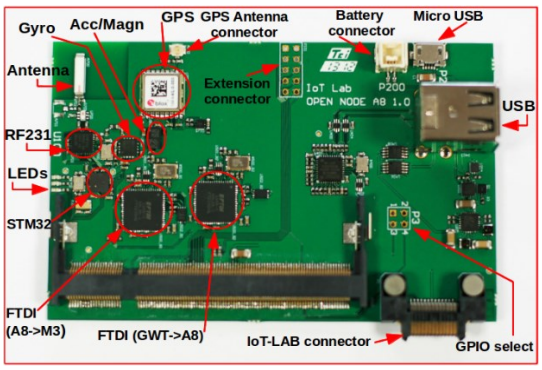
- from any other embedded Linux board currently running,
- from any board of the testbed based on a microcontroller having a public IPv6 address,
- from any IoT-LAB SSH frontend,
- and, on the whole, from any public IPv6 address on the Internet.

Each site has a /64 IPv6 prefix, in which each embedded Linux node will get its static IPv6 address. Its IPV6 configuration can be known with the following environment variables:

```
root@node-a8-1:~# printenv
...
GATEWAY6_ADDR=2001:660:4701:f080:ff::
INET6_ADDR=2001:660:4701:f080::1/64
```

IoT-LAB A8-M3

The IoT-LAB A8-M3 board is based on a TI SITARA AM3505 which is a high-performance ARM Cortex-A8 microprocessor.

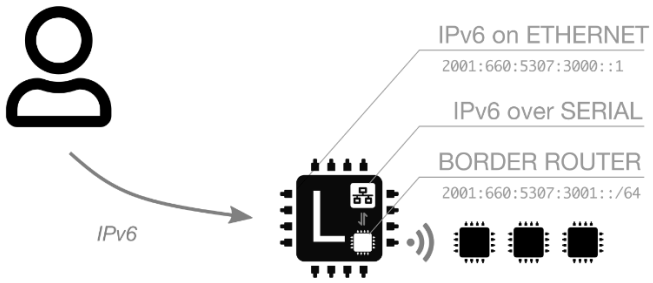


The IoT-LAB A8-M3 embed a clone of the IoT-LAB M3 object (M3 co-microcontroller). As a result, it features the same sensors/actuators, as well as the same radio chip, enabling it to communicate wirelessly with other 802.15.4 objects within radio range. The M3 is linked to the A8 via the I2C data bus and the GPIO inputs/outputs. This M3 co-microcontroller can be programmed from the A8 through a USB/FTDI component (see Programming).

Mixing two types of boards

It is possible to test common IoT scenarios that implies communications between sensors and an application gateway, the later needing to have a more powerful environment. Boards based on a microcontroller play the role of sensors and a board running embedded Linux, equipped with a co-microcontroller managing a radio interface, plays the role of the gateway.

In that case, it's therefore quite possible to build an IPv6 network with the co-microcontroller acting as a Border Router and communicating with other nodes by radio (e.g. 802.15.4). Unlike the setup with a microcontroller on the SSH frontend, the creation of the virtual network interface and IPV6 traffic encapsulation is done directly on the embedded Linux node using the co-microcontroller serial's link.



Ethernet

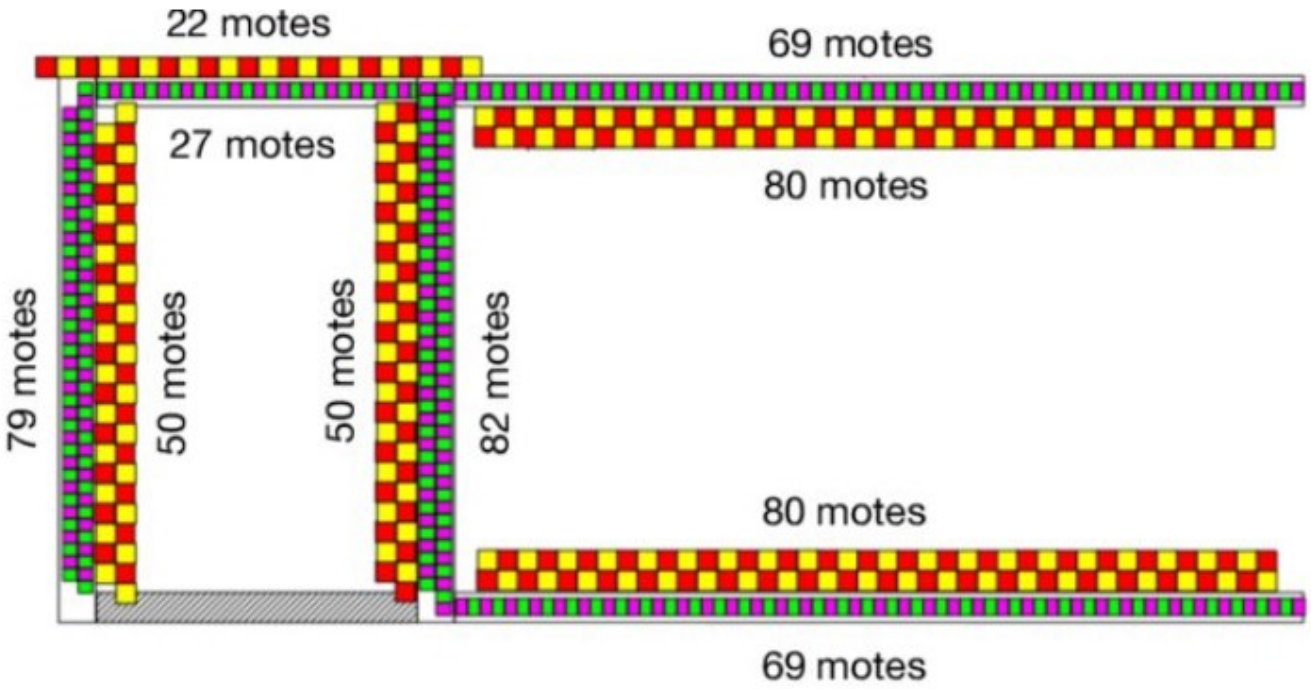
The IoT-LAB A8-M3 board features an Ethernet interface, enabling it to connect to a LAN and to communicate with the internet. The boards are only accessible by SSH in IPV4 via the SSH frontend and in IPV6 from the Internet.

How is the deployment of multiple sensors going to affect the IoT platform?

The platform is equipped with m3 boards which have four types of sensors.

- Ambient sensor light - ISL29020
- Atmospheric pressure and temperature - LPS331AP
- Tri-axis accelerometer / magnetometer - L3G4200D
- Tri-axis gyrometer - LSM303DLHC

These sensors measure environmental variables such as pressure, temperature, light. Since the grenoble site has the layout described in the next figure, it can be deduced that the variables measured by the sensors on different devices should not vary too much each from the others.

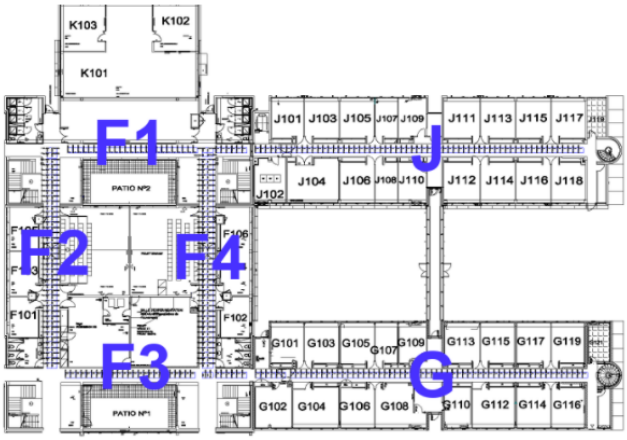


Topology

Grenoble testbed is deployed at a building scale, over the ground floor of Inria building, through corridors in the raised floors and dropped ceilings.

- The nodes in the raised floors are fixed horizontally under the tiles.
- The nodes in the dropped ceilings are fixed vertically to the wall at a height of 2.6 m and 3.2 m.

Here is the distribution of nodes by corridor:



- **J corridor**
  - floor: `m3-[1-69]`
  - ceiling: `m3-[359-380]`, `a8-[1-58]`
- **F1 corridor**:
  - floor: `m3-[70-94]`
  - ceiling: `a8-[59-77]`
- **F2 corridor**:
  - floor: `m3-[95-178]`
- **F3 corridor**:
  - floor: `m3-[179-205]`
  - ceiling: `a8-[78-94]`
- **F4 corridor**:
  - floor: `m3-[206-289]`
  - ceiling: `a8-[95-148]`
- **G corridor**:
  - floor: `m3-[290-358]`
  - ceiling: `a8-[149-228]`

Any variations that may occur could be due either to aerators, chillers, air conditioners, radiators, local overheating, or measurement errors. In some cases the sensors can also measure erroneously, or not be calibrated and therefore lead to even large measurement variations. The types of errors coming from sensors are many and they are not the only ones. In other cases there may be network communication errors, leading to missing readings. For example, it can happen that a board disconnects, or rhymes, and consequently the sensors on it will not take readings. or it may happen that the sensors themselves break, or have become obsolete. In all these cases the solution can be the redundancy of the equipment or temporal redundancy. That is, double the readings or devices.

It was decided to create a mesh network of m3 nodes (embedded on A8 nodes or standalone) connected to each other with IEEE 802.15.4 standard, used by the Zigbee communication protocol. Each node uses Public IPv6 (6LoWPAN / RPL) network with M3 nodes in IOT Lab.

Differences between a Wireless Sensor Network (WSN) and Internet of Things

The Internet of Things, commonly abbreviated as IoT, refers to the connection of devices to the Internet.

INTERNET of Things (IoT) is a network of connected objects, each with low storage, limited energy, and processing capabilities. These objects interact in a complex way to enhance reliability, performance, and security of their infrastructure.

In an IoT system, all of the sensors directly send their information to the Internet. For example, a sensor may be used to monitor the temperature of a body of water. In this case, the data will be immediately or periodically sent directly to the internet, where a server can process the data and it can be interpreted on a front-end interface.

Conversely, in a WSN, there is no direct connection to the Internet. Instead, the various sensors connect to some kind of router or central node. Then route the data from the router or central

node to Internet. That being said, an IoT system can utilize a wireless sensor network by communicating with its router to gather data.

You can think of a wireless sensor network (WSN) as more of a group of sensors or "a big sensor" and less like a "competitor" or "rival" to the Internet-of-Things.

In this sense WSN is as a Subset of IoT.

IoT exists at a higher level then WSN. In other words, WSN is often a technology used within an IoT system.

It is important to note that a large collection of sensors, as in a mesh network, can be used to individually gather data and send data through a router to the internet in an IoT system.

It's also important to note that the term "wireless sensor network" is not nearly as encompassing as "the internet of things." WSN consists of a network of only wireless sensors. If the network was to include a wired sensor, it could no longer be labeled a "wireless sensor network." This is unlike IoT. Essentially any device that connects to the Internet can be considered an IoT device. An "IoT

system" can therefore be interpreted as a group of many IoT devices.

We conclude that it is true the following definition of WSN.

Wireless Sensor Network - A collection of wireless sensors that may or may not be connected to the internet.

Experiments to be created in the IOT Lab Testbed

It is going to be created an experiment in IOT Lab testbed with four nodes. Three of which are M3 nodes (M3-10,M3-11,M3-12), , and three of which are A8 nodes. One of the M3 nodes will act as a gateway or border router (BR). The other two will be simply wireless sensor nodes.
This will be an M3 simple mesh network with a BR nodes interconnecting he other nodes to the Internet.
The commands to build this M3 network will be the following:

(Taken from https://www.iot-lab.info/legacy/tutorials/riot-public-ipv6-m3/index.html)

1) create from the web interface dashboard 3 M3 nodes, with their profiles (traffic and performances), and 3 A8-M3 nodes

2) $ ssh bruzzese@saclay.iot-lab.info
3) iotlab-auth -u bruzzese (not needed
4) $ mkdir -p ~/riot
5) cd ~/riot
6) Rm -rf RIOT
7) $ git clone https://github.com/RIOT-OS/RIOT.git -b 2020.10-branch
8) $ cd RIOT
9) $ source /opt/riot.source
10) $ make ETHOS_BAUDRATE=500000 DEFAULT_CHANNEL=20 BOARD=iotlab-m3 -C examples/gnrc_border_router clean all
11) $ iotlab-node --flash examples/gnrc_border_router/bin/iotlab-m3/gnrc_border_router.elf -l saclay,m3,10
12) $ sudo ethos_uhcpd.py m3-10 tap0 2001:660:3207:04c1::1/64

In another terminal, in the front end

13) $ ssh bruzzese@saclay.iot-lab.info
14) $ ip -6 route

Open now another terminal and issue the following commands :

1) $ ssh bruzzese@saclay.iot-lab.info
2) $ cd riot/RIOT
3) $ source /opt/riot.source
4) $ make DEFAULT_CHANNEL=20 BOARD=iotlab-m3 -C examples/gnrc_networking clean all
5) $ iotlab-node --flash examples/gnrc_networking/bin/iotlab-m3/gnrc_networking.elf -l saclay,m3,11
6) $ iotlab-node --flash examples/gnrc_networking/bin/iotlab-m3/gnrc_networking.elf -l saclay,m3,12

In another terminal, in the front end

1) $ ssh bruzzese@saclay.iot-lab.info
2) $ nc m3-11 20000
3) > ifconfig
4) > udp server start 8888
5)

The global prefix has been successfully propagated and the IP on the M3 can be known with the ifconfig command. We can verify that it answers to “ping” from the frontend SSH (and from any M3 node of the experiment with this networking installed on it.

We also start an UDP server and send a message to an M3 node address.

Open now another terminal and issue the following commands :

1) `$ ssh bruzzese@saclay.iot-lab.info`

2)
```
$ echo "hello" > /dev/udp/2001:660:3207:4c1:1711:6b10:65fd:bd36/8888
```

The commands to build this A8-M3 network will be described in the the following lines.
First of all it must be built the required firmware for the border router node. The node A8 will act as the border router in this experiment. The border firmware is built using the RIOT gnrc border router example
.This node-A8 will be responsible for propagating an IPv6 global prefix and assign an address to the device itself.

The border router will route packets between a 6Lo network (PAN) and a 'normal' IPv6 network (i.e. the Internet).

This requires the border router to have two interfaces: A downstream interface to run 6LoWPAN on and an IPv6 uplink.

The IPv6 traffic of the Border Router (BR) to ethernet is then encapsulated on the BR's serial link and routed via a virtual network interface (tap).

To connect to the serial link of your border router and propagate an IPv6 prefix through your network, RIOT provides the ethos_uhcpd tool. It uses the serial interface, ethos (Ethernet Over Serial) and UHCP (micro Host Configuration Protocol). Ethos multiplexes serial data to separate ethernet packets from shell commands. UHCP is in charge of configuring the wireless interface prefix and routes on the Border Router.

The script "start_network.sh" enables a *ready-to-use* BR in only one command.

The following commands will build the firmware of A8 node BR:

1. create from the web interface dashboard 3 A3 nodes, with their profiles (traffic and performances)
2. $ ssh rbruzzes@grenoble.iot-lab.info
3. $ iotlab-auth -u rbruzzes
4. $ iotlab-experiment submit -n riot_a8 -d 120 -l 3,archi=a8:at86rf231+site=Grenoble (only if not 1)
5. $ mkdir -p ~/A8/riot
6. $ cd ~/A8/riot
7. $ rm -rf RIOT
8. ~~$ git clone https://github.com/RIOT-OS/RIOT.git -b 2020.10-branch~~
9. $ git clone https://github.com/RIOT-OS/RIOT.git -b 2021.01-branch
10. $ cd RIOT
11. $ source /opt/riot.source

    We build the BR and GNRC networking
12. $ make ETHOS_BAUDRATE=500000 DEFAULT_CHANNEL=19 BOARD=iotlab-a8-m3 -C examples/gnrc_border_router clean all
13. $ cp examples/gnrc_border_router/bin/iotlab-a8-m3/gnrc_border_router.elf ~/A8/
14. $ make DEFAULT_CHANNEL=19 BOARD=iotlab-a8-m3 -C examples/gnrc_networking clean all
15. $ cp examples/gnrc_networking/bin/iotlab-a8-m3/gnrc_networking.elf ~/A8/

    We flash the BR firmware on the first node A8-110
16. $ ssh root@node-a8-110
17. $ ~~iotlab_flash A8/gnrc_border_router.elf~~
18. $ flash_a8_m3 A8/gnrc_border_router.elf
19. $ cd ~/A8/riot/RIOT/dist/tools/uhcpd
20. $ make clean all
21. $ cd ../ethos
22. $ make clean all
23. printenv

(Take inet6_prefix that is like 2001:660:5307:306e)

24. $ reset_a8_m3
25. $ *ip -6 r (see if some tap or address is busy*

    The answer of the system will be like the following:

    2001:660:5307:3000::/64 dev eth0 proto kernel metric 256

    fe80::/64 dev eth0 proto kernel metric 256

    ff00::/8 dev eth0 metric 256

    default via 2001:660:5307:3000:ff:: dev eth0 metric 1024

    We configure the network on the BR
26. $ ./start_network.sh /dev/ttyA8_M3 tap0 2001:660:5307:306e::1/64 500000

    We flash the gnrc_networking firmware on another M3 node
27. $ ssh rbruzzes@grenoble.iot-lab.info
28. $ ssh root@node-a8-115
29. $ iotlab_flash A8/gnrc_networking.elf

Or do the following command

```
login@grenoble:~/riot/RIOT/$ iotlab-node --flash
examples/gnrc_networking/bin/iotlab-m3/gnrc_networking.elf -l grenoble,m3,115
```

This firmware let communicate different instances of RIOT or between RIOT instance and Linux Host (as in the A8-M3 node).

It should be installed in all the M3 nodes that needs to be connected.

https://www.iot-lab.info/learn/tutorials/riot/riot-public-ipv6-m3/

https://www.iot-lab.info/legacy/tutorials/riot-networking-m3/index.html

se vedi https://www.iot-lab.info/legacy/tutorials/riot-networking-m3/index.html

con un semplice gnrc_networking i nodi M3 sono raggiungibili tra loro con un ping6.

Cioè c'è una rete wireless.

L'esempio https://www.iot-lab.info/legacy/tutorials/riot-public-ipv6-m3/index.html

funziona perfettamente.

The IoT-LAB A8-M3 embed a clone of the IoT-LAB M3 object. As a result, it features the same sensors/actuators, as well as the same radio chip, enabling it to communicate wirelessly with other 802.15.4 objects within radio range. The M3 is linked to the A8 via the I2C data bus and the GPIO inputs/outputs. This M3 co-microcontroller can be programmed from the A8 through a USB/FTDI component.

We will do the same on the other M3 nodes.

We can then interact with the M3 nodes where we installed `gnrc_networking` m3-2 using nc.

6LoWPAN is an acronym of *IPv6 over Low -Power Wireless Personal Area Networks*.[1] 6LoWPAN is the name of a concluded working group in the Internet area of the IETF.[2]

The 6LoWPAN concept originated from the idea that "the Internet Protocol could and should be applied even to the smallest devices,"[3] and that low-power devices with limited processing capabilities should be able to participate in the Internet of Things.[4]

The 6LoWPAN group has defined encapsulation and header compression mechanisms that allow IPv6 packets to be sent and received over IEEE 802.15.4 based networks

```
my_computer$ ssh <login>@grenoble.iot-lab.info
login@grenoble:~$ nc m3-2 20000
```

We can now have access to the RIOT shell and can ping another Internet host using IPv6 (let's try a Google DNS host):

```
> ping6 2001:4860:4860::8888
```

Use RIOT shell ifconfig command to get the IP of the M3 node:

```
> ifconfig
Iface  7   HWaddr: 29:02  Channel: 26  Page: 0  NID: 0x23
        Long HWaddr: 36:32:48:33:46:df:a9:02
        TX-Power: 0dBm  State: IDLE  max. Retrans.: 3  CSMA Retries: 4
        AUTOACK  CSMA  MTU:1280  HL:64  6LO  RTR  RTR_ADV  IPHC
        Source address length: 8
        Link type: wireless
        inet6 addr: ff02::1/128  scope: local [multicast]
        inet6 addr: fe80::1711:6b10:65fd:bd36/64  scope: local
        inet6 addr: ff02::1:fffd:bd36/128  scope: local [multicast]
        inet6 addr: 2001:660:3207:4c1:1711:6b10:65fd:bd36/64  scope: global
        inet6 addr: ff02::2/128  scope: local [multicast]
```

The global prefix has been successfully propagated, the IP on the M3 is 2001:660:3207:4c1:1711:6b10:65fd:bd36. Verify that it answers to “ping” from the A8 (and from any computer with a global IPv6):

```
login@grenoble:~$ ping6 -c 3 2001:660:3207:4c1:1711:6b10:65fd:bd36
PING 2001:660:3207:4c1:1711:6b10:65fd:bd36(2001:660:3207:4c1:1711:6b10:65fd:bd36) 56
data bytes
64 bytes from 2001:660:3207:4c1:1711:6b10:65fd:bd36: icmp_seq=1 ttl=61 time=45.7 ms
64 bytes from 2001:660:3207:4c1:1711:6b10:65fd:bd36: icmp_seq=2 ttl=61 time=46.5 ms
64 bytes from 2001:660:3207:4c1:1711:6b10:65fd:bd36: icmp_seq=3 ttl=61 time=44.9 ms

--- 2001:660:3207:4c1:1711:6b10:65fd:bd36 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 44.919/45.759/46.595/0.684 ms
```

Still on m3-2, let’s verify that UDP packets can be received from the SSH frontend. Use the udp command in the gnrc_networking shell to start an UDP server on the M3 node:

```
> udp server start 8888
udp server start 8888
Success: started UDP server on port 8888
```

Then from the SSH frontend, send a message by udp:

```
login@grenoble:~$ echo "hello" > /dev/udp/2001:660:3207:4c1:1711:6b10:65fd:bd36/8888
```

The packet will be received on M3 node.

Segue https://www.iot-lab.info/learn/tutorials/riot/riot-public-ipv6-a8-m3/

after carrying out the propagation of the ipv6 addresses, each node will be reachable on the internet with its global ipv6 address.

https://ieeexplore.ieee.org/document/7868374

https://ieeexplore.ieee.org/document/7917094

- Consider the benefits of deploying multiple sensors with overlapping observation areas in terms of data quality, fault tolerance, and energy efficiency.

The benefits from the point of view of data quality, fault tolerance, and energy efficiency of having multiple sensors that have overlapping observation areas will be evaluated.

Data Quality

The sensors are the main source of data. *Errors* in sensing measurements can be handled by procedures that are established based on a deep understanding of the characteristics of the sensors. *Missing readings* may be handled by oversampling, and glitches, like *outliers and noise*, can be masked by averaging.

System dependability

The accuracy of data means its quality. The concept of dependability of a system is strictly related to the data quality.

Dependability can be defined as the ability of a system to avoid service failures that are more frequent or more severe than is acceptable.

The problem becomes critical when dependability is an important application requirement. For instance, in water-related information systems, inaccurate information in aquatic monitoring may lead to false warnings being issued or harmful situations not being detected early enough (e.g., floods or pollution events). As another example, WSNs are deployed in data centers for flexible temperature monitoring and energy-efficient control of air-cooling equipment.

Therefore, ensuring the accuracy of collected data is also necessary for effectiveness reasons. In these examples, the operational conditions are typically hard to accurately predict, ensuring that the reliability of operations is often hard or costly, and the consequences of inaccurate sensor data collection can be detrimental.

The root cause of dependability problems concerning the quality of sensor data, that is the several kinds of faults that may affect the system operation, in particular at the sensor and network levels, describing the specific effect on the sensor data and the relevant failure modes-

It is important to know that fault-tolerance strategies based on sensor data fusion procedures, exploiting the availability of redundant measurements or available modeling surrogates, do exist. The collected data should be continuously validated , since sensors require maintenance.

Faults at the sensor level

As far as concerns individual sensor measurements, the possible types of errors observed in measurement values can be classified as follows:

1. **Random errors** are described by an absence of repeatability in the readings of the sensor, for instance due to measurement noise. These errors tend to happen on a permanent basis, but have a stochastic nature;
2. **Systematic errors** are described through consistency and repeatability in the temporal domain. There are three types of systematic errors at the sensor level: –
3. **Calibration errors** result from errors in the calibration [iii]procedure, often in relation to linearization procedures; –
4. **Loading errors** emerge when the intrusive nature of the sensor modifies the measurand. Along with calibration errors, loading errors are caused by internal processes; –
5. **Environmental errors** emerge when the sensor experiences the surrounding

environment and these influences are not considered. In contrast with the previous two types of errors, environmental errors are due to external factors;

6. **Spurious readings** are non-systematic reading errors. They occur when some spurious physical occurrence leads to a measurement value that does not reflect the intended reality. For instance, a light intensity measurement in a room can provide the wrong value if obtained precisely when a picture of the room is taken and the camera flash is triggered.

Sensor Redundancy and Sensor Fusion

Redundancy implies that the fault has been understood and detected. Redundancy does not refer solely to having multiple similar components as in our case of multiple and contiguous M3. It is also possible to implement forms of redundancy in the time (e.g., repeating some action multiple times).

Providing the system with redundant components, that means redundant sensors, can compensate

existing errors or faults affecting some component. The affected component can be replaced in its tasks by the spare, redundancy component, which will ensure that the system function will continue to be provided.

Sometimes sensor fusion techniques can be applied. While in single-sensor situations, there are models or related information that allow reasoning about an individual sensor's data quality without requiring other sensors' data, in sensor fusion techniques the process of combining sensory data or data derived from disparate sources is adopted such that the resulting information has less uncertainty than would be possible when these sources were used individually.

That means that , in the case of M3 board and sensors, we could correlate temperature with light sensor , in order increase data quality and reliability.

In a multi-sensor situation, the quality of sensor measurements is characterized by using redundant or correlated data obtained from the different sensors. This redundancy allows for data fusion methods to be deployed at the network level,

resulting in improved (fused) sensor data, as well as improved data quality characterization.

Fault detection methods

A fault detection method for understanding sensor operation can be based on :

1. Rule-based methods that use expert knowledge about the variables that sensors are measuring to determine thresholds or heuristics with which the sensors must comply.
2. Estimation methods that define a "normal" behavior by considering spatial and temporal correlations from sensor data. A sensor reading can be matched alongside its forecasted value to assess its validity.
3. Learning-based methods that define models for correct and faulty sensor measurements, using collected data for building the models.
4. Fuzzy logic rules to obtain a qualitative sense of a sensor's validity based on its

own historical behavior represented by a confidence measure.

Sensor Failure Modes

The main sensor failures modes, are the following :

1. **Constant or offset failure mode:** The observations continuously deviate from the expected value by a constant offset.
2. **Continuous varying or drifting failure mode:** The deviation between the observations and the expected value is continuously changing according to some continuous time-dependent function (linear or non-linear).
3. **Crash or jammed failure mode:** The sensor stops providing any readings on its interface or gets jammed and stuck in some incorrect value.
4. **Trimming failure mode:** The observations are correct for values within some interval, but are modified for values outside that interval. Beyond the interval, the observation can be trimmed at the interval boundary or

may vary proportionally with the expected value.

5. **Outliers failure mode:** The observations occasionally deviate from the expected value, at random points in the time domain;
6. **Noise failure mode:** The observations deviate from the expected value stochastically in the value domain and permanently in the temporal domain-

Faults at the network level

When connecting individual sensor nodes in a wireless sensor network, additional faults affecting sensor data can be introduced by the network. The main kinds of network faults that may affect the quality of sensor data in order to achieve a reliable network operation, specifically considering faults in the time domain and faults in the value domain. In the time domain, a crash, omission or delay faults could occur.

1. **Crash faults** (for instance of the radio subsystem in a sensor node) lead to data

absence and can only be mitigated with redundancy (e.g., a dual-radio system).

2. **Omissions** correspond to missing sensor readings due to lost messages. They can be prevented by enforcing communication reliability, for instance based on message retransmission. However, reliable communication protocols are not very common in WSNs due to the additional resources (namely energy) they require. Therefore, omissions do happen in sensor networks and for the most part emerge because of sensor failures and packet losses.

   Heavy packet loss and asymmetric links occur frequently in WSNs [32,33], for instance due to signal strength fading and intermittent or continuous environmental interference (e.g., wind or rain).

3. **Attacks** that may significantly affect the quality of sensor data, among other

consequences for the application. In critical applications, it is important to deploy security techniques to avoid attacks or to mitigate their effects.

However, communication protocols typically incorporate data integrity verification mechanisms that allow the detection of corrupted messages, discarding those messages and hence transforming value faults into omission faults. Therefore, the only chance that received data do not correspond to what has been sent is when some part of the communication stack in the sending or receiving node (or both) is affected by an accidental fault not covered by the integrity verification mechanisms or when it has been intentionally corrupted

The data processing level

Finally, in order to centrally analyze all of the sensing information, a third layer appears in the system. In the data processing layer, it is possible to infer the quality of the gathered information, through fusion processes of redundant and related measurements, by multi-sensor fusion methods, or by expert-knowledge of the system model. In fact, this is where

we can ultimately handle both sensor and network-level problems and apply some mitigation techniques.

Energy Efficiency

Obviously the more the redundancy, the more will be the energy consumption , since there will be multiple sensor measurements, multiple data processing, multiple transmissions.

Especially for large-scale deployment since sensors nearby the gateway more often consume energy and deplete earlier, or may face temporal death and disconnect from the network, which makes the lifetime of the whole network short.

Energy Consumption Constraint: The IoT energy consumption model depends on expenditure in transmit and acquisition, while processing and sensing expenditures are less than data transmission/receiption. Selecting the subsequent hop is achieved via exploiting the nearest neighbor, and therefore each sensor has a transmission range to communicate with neighbors.

- Identify the technical limitations of a multi-hop wireless network in terms of throughput, end-to-end delay and security.

In the kind of mesh network we are being setting up , in order to maintain the connectivity, the sensor nodes should be appropriately deployed. We can arrange a deployment by selecting the nodes of our topology. The cost of transmitting a message among sensors is independent of the number of receiving sensors, however, it depends on a function of their maximum distance for sending a message from sensors. Therefore, in the case of multihop networking, then it is possible to maintain connectivity without every sensor transmitting at maximum power. This allows us to seek optimal power range assignment for connectivity and other related network issues.

The capacity of a network is an important parameter for evaluating the performance of both an electronic device and a telecommunications system; in the latter case, the transmission capacity is simply the

maximum transmission speed of the line. For a better evaluation of the capacity of a network, reference is made to the **throughput**, i.e. the amount of information conveyed on a channel at a given time, which is always less than or equal to the maximum capacity and depends exclusively on how much information is introduced on the channel during the transmission; the throughput is therefore a specification of the transmission that relates to the maximum capacity of the network, which instead is fixed, constant.

As far as concerns **end-to-end delay** Constraint it is accounted for by finding the optimal number of hops in a path which might lead to different delay guarantees in the network. This delay might be classified into several types as queuing, propagation, processing, transmission, retransmission, and idle. The delay will be increasingly proportional to the number of hops- and will result by the product of the number of hops by the sum of all the componens of the single delay (Dqueu + Dprop + Dproc + Dtrans + Dretrans + Didle).

Besides energy is correlated with delay. For example, a smart meter requires a higher energy efficiency but has a low requirement for delay. Meanwhile, a vehicular network is rather delay sensitive. Therefore, it is essential to understand the mechanism of controlling the neighboring set of sensors in different network technologies by adjusting the transmission range and/or selecting appropriate sensors to carry out a specific task

As far as concern **security** the multihop topology is a target of many possible attacks, since it is using the RPL protocol (see Figure x).

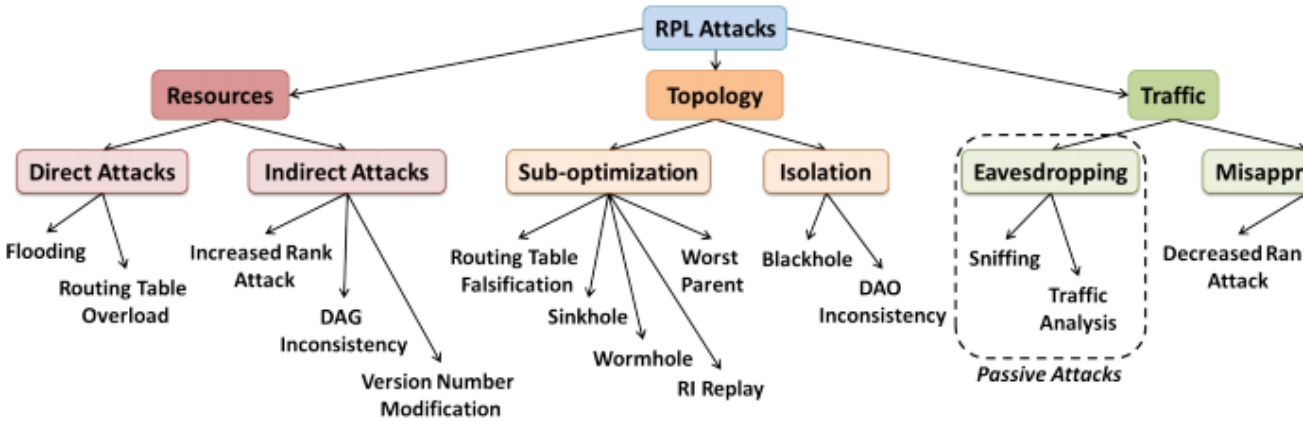


Figure 3.1: Taxonomy of attacks against RPL networks.

As it can be seen from the Figure above, there exist a plethora of attack to this kind of mesh networks.

Just to consider one of them, the traffic type of attack, or eavesdropping , which is passive type in nature.

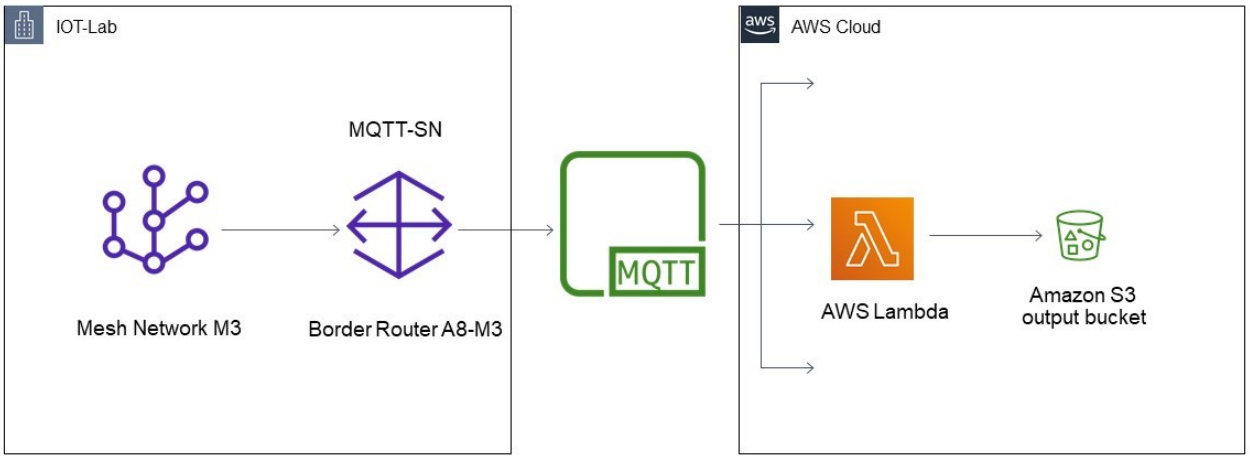
A sniffing attack consists in listening to the packets transmitted over the network. This attack is very common in wired and wireless networks and compromises the condentiality of communications. An attacker can perform this attack using a compromised device or directly capture the packets from the shared medium in case of wireless networks. The information obtained from the sniffed packets may include partial topology, routing information and data content. In RPL networks, if an attacker sniffs control messages, it can access information regarding the DODAG configuration such as DODAG ID, version number, ranks of the nodes located in the neighborhood. By sniffing data packets, the attacks can not only discover packet content but also have a local view of the topology in the eavesdropped area by looking at source/destination addresses. This attack is difficult to be detected due 38 Chapter 3.

Taxonomy of Attacks in RPL Networks to its passive nature. The only way to prevent sniffing is encryption of messages when the attacker is external. Even if RFC 6550 mentions encryption of control messages as an option, the technical details are left out from the specification making implementation difficult.

https://www.iot-lab.info/legacy/tutorials/monitoring-sniffer-m3/index.html

What are the connected components, the protocols to connect them and the overall IoT architecture?

- Provide a **new** network diagram that includes the wireless sensor network and the new network technologies and communication protocols used to interconnect it with the cloud-based elements.



- Identify any additional software components required to that make up your system both at IoT device level and at cloud level.
-
- Provide a **new** high-level architecture diagram that depicts the interdependencies of your software components.

1. *How do you measure the performance of the system?*
    - Measure the performance of the wireless sensor network. Measure the end-to-end latency and throughput. Measure the overall quality of the wireless communication channel.

We have already defined the troughput. The definition of latency is based on ping, defined as the

time interval between the time when an input is sent and when the output signal is received. It is the delay in data transmission over a network, that is: the response speed of a system to a pulse. The average time in milliseconds spent processing end to end message transactions over the selected time frame.
We could then measure it as the average response time from terminal nodes M3 to ping border router A3.

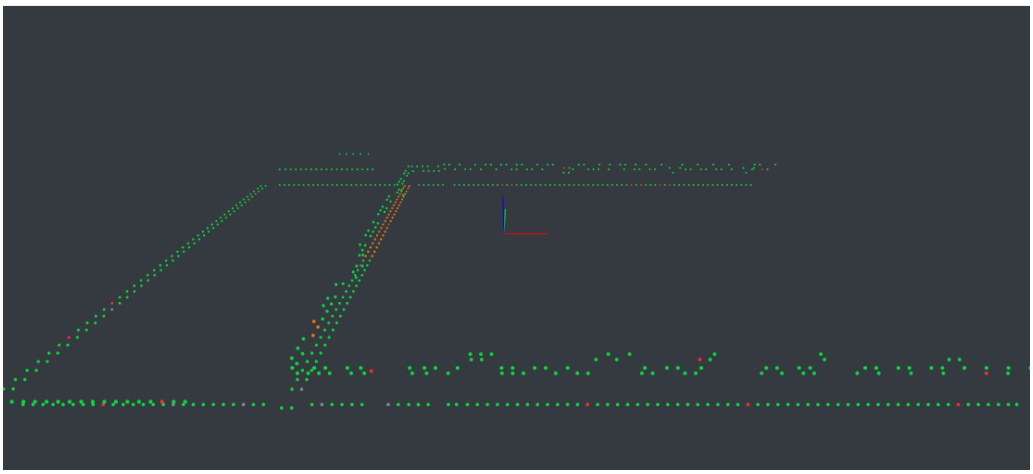
As far as concerns quality , in theory there are many sensors nodes and border router nodes to be distributed in a the Grenoble site area, however, the goal of deploying should be to determine the optimal number of hops between the sensors nodes and the border router nodes while satisfying the quality (QoS) parameters. The model is presented as an optimization problem for quality in terms of energy consumption, delay, and throughput

The quality of service (QoS) or quality of network can be defined in terms of energy consumption, delay, and throughput of the network.

For having a better quality of the network the throughput should be increased , while minimizing the energy consumption and the delay,

In order to be more precise for Iot systems to be successful operations presume the qualities of the 5 experience definition based on energy efficiency, connectivity, throughput, availability, and delay sensitivity.

i



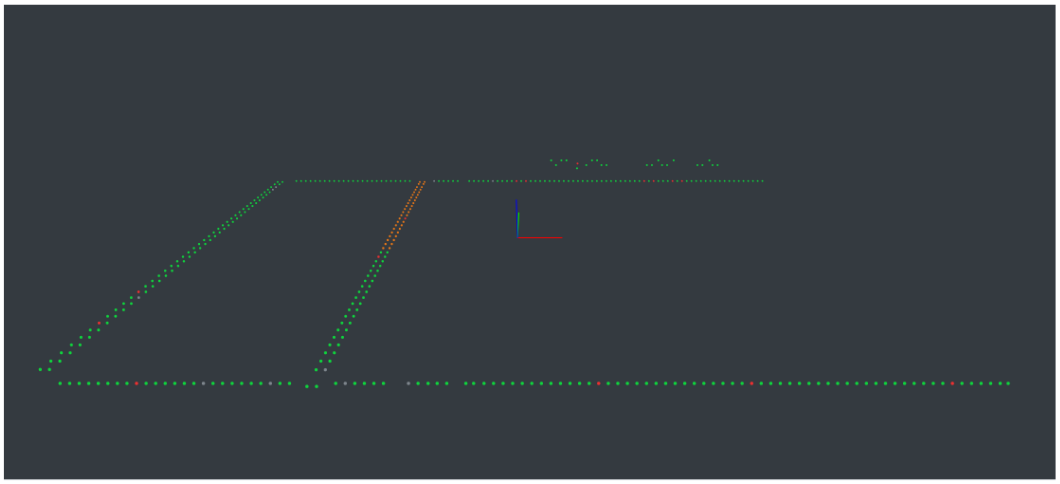
It has been chosen grenoble site as the site to carry out the experiment because it seemed to be available and also from the interesting topology. In fact it has a long dislocation of the corridors and is also

configured in a horseshoe shape. There are boards both on the floor and on the ceiling. For the purposes of the experiment, as I do not know the exact distances and lengths or widths of the various rooms, and also if there are obstacles between the rooms, such that the wireless network signal may not reach the devices. But they are not, it would be interesting to know if the various environments have different climatic conditions, different light exposures and so on. This becomes a tool for the investigation of physical environments done remotely just as a wsn network allows us to know a territory in its environmental variables.

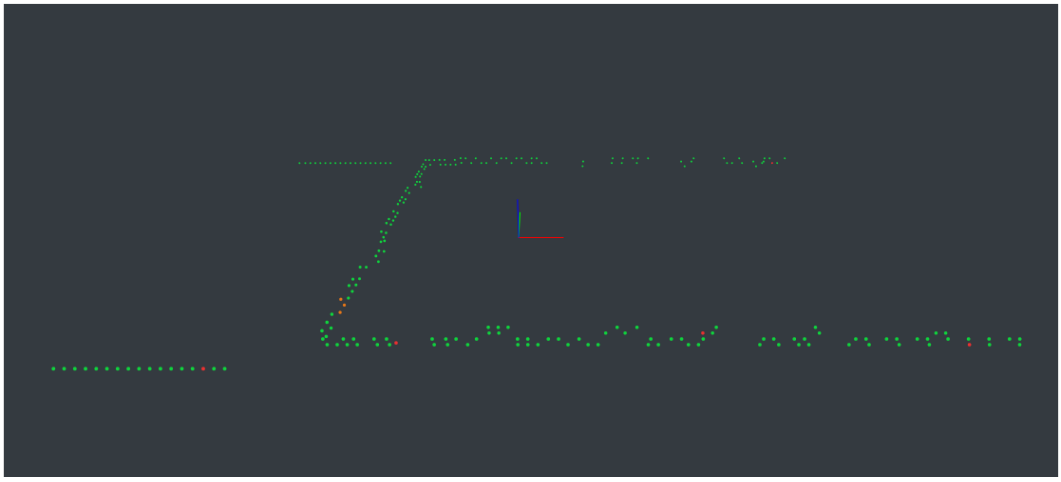
You can also make deductions on the distances between the rooms as we know the range of action of 802.15.4 wireless networks. We know that

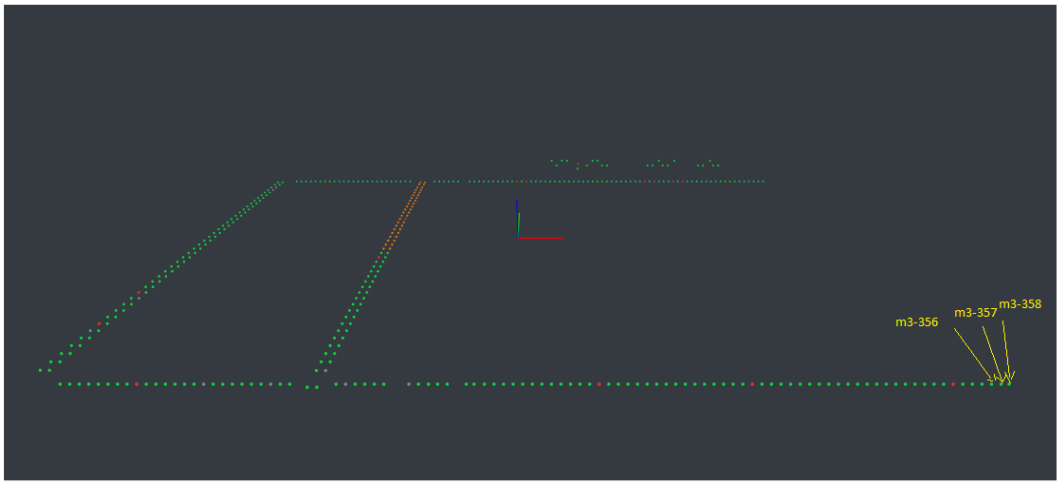
In the case of **6LoWPAN** networks, the distance is 10 –100 m.

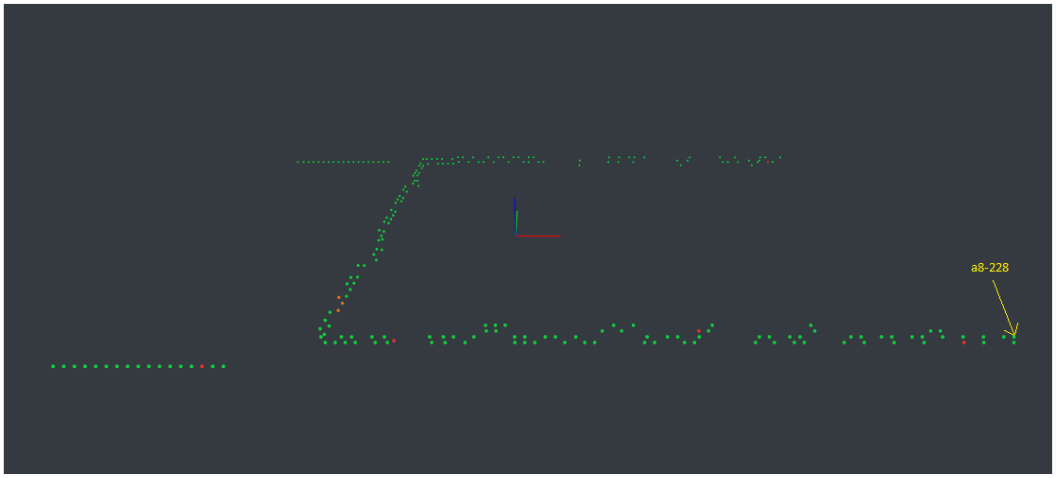
This is m3 deployment



This is the a8 deployment







as you can see, the deployment of m3 and a8 are almost superimposed, except for corridor F2 where there are no nodes a8 but only m3.

Since the role of the A8 nodes will be that of border router, this means that the m3 nodes. Which are generic sensor nodes, in this corridor will necessarily be distant from the relative border router.

A question arise about the connectivity of these nodes under our wireless sensor network.

The way I intend to build the sensor wireless network is by assigning and propagating an ipv6 address to each board, and using the 6Lowpan network protocol. I mean Public IPv6 (6LoWPAN /

RPL) network with M3 nodes. I decided to do three different experiments to check the quality parameters of wireless sensor networks.

The first experiment I intend to carry out using three m3 nodes, in the role of application sensor nodes, and a node a8, having the role of boarder router, located contiguously. In this case the diameter (the maximum distance between two nodes in the **network**) of the realized network will be a few centimeters, and the overall performance of the wireless network, the network latency, the output, the overall quality of the communication channel will be measured.

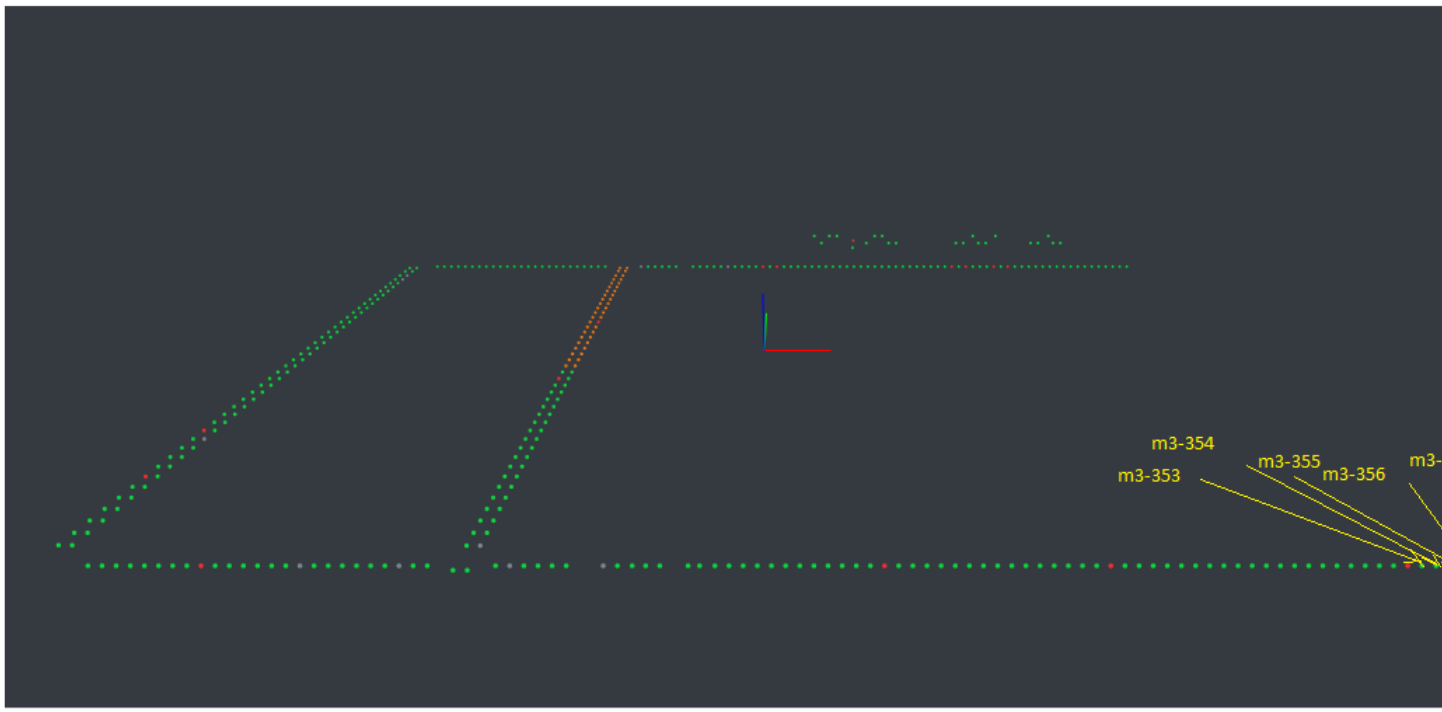
The m3 nodes will be : m3-358, m3-357, m3-356; while the a8 node will be a8-228.

Energy consumption and duty cycle will be measured in relation to the performance of the wireless network. The negative aspects and limits of multi hop will be evaluated from the point of view of throughput, delay, safety.

In order to evaluate the performance of the system as the number of wireless elements increases i have added to the previous topology three more m3

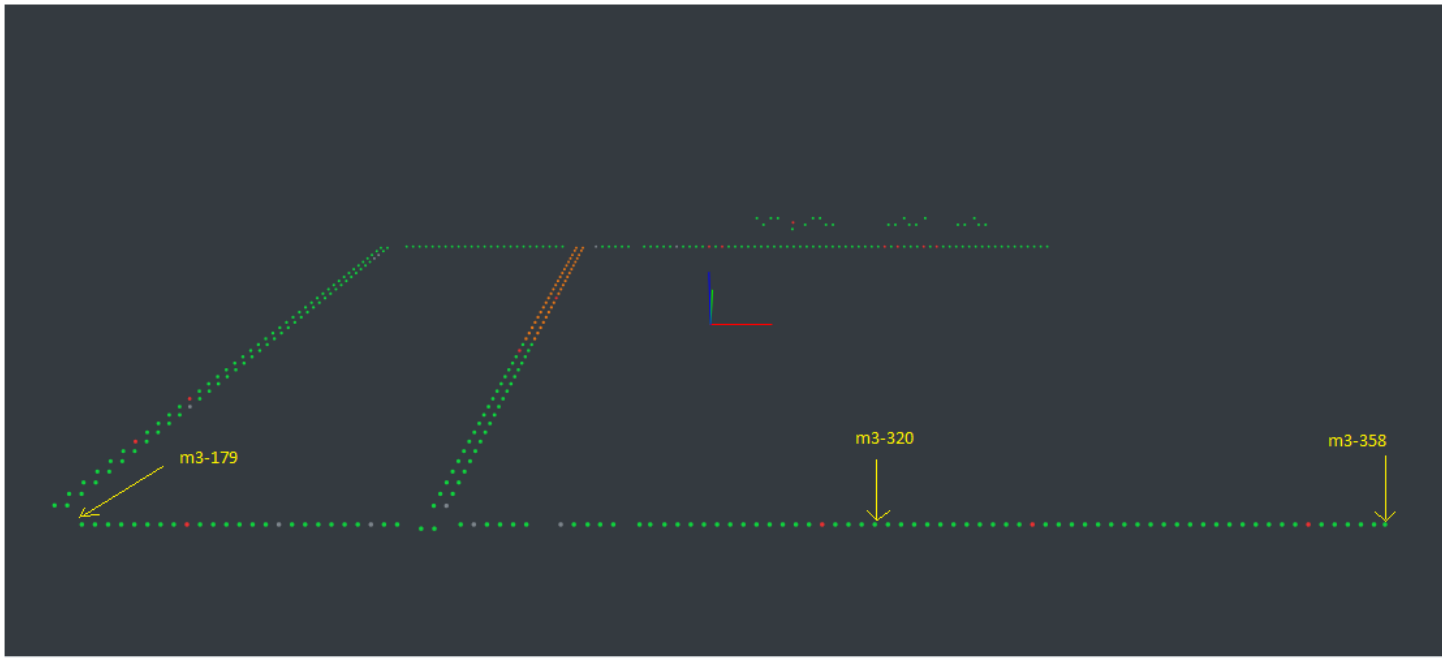
nodes, contiguous to the other ones. In this case the diameter of the wireless sensor network is still very small, the order of some centimeters.

Let us see the graphics of the topology.



In order to assess How the physical location of the nodes affect the performance of the wireless network i have modified the topology of experiment 1 in order to move two of the three sensors in the middle of corridor G and on the extreme opposite site in the corridor F3. Although it cannot be known the exact distance among the nodes fo this topology, it should be withing 100 meters. Anyway some obstacles

could be in the path and also it must be known that 10-100 metres is right the maximum distance reachability of 6lowpan network nodes. This experiment has been set in order to measures how much variation in the performances will the diameter variation affect the overall performances of the wireless sensor network.



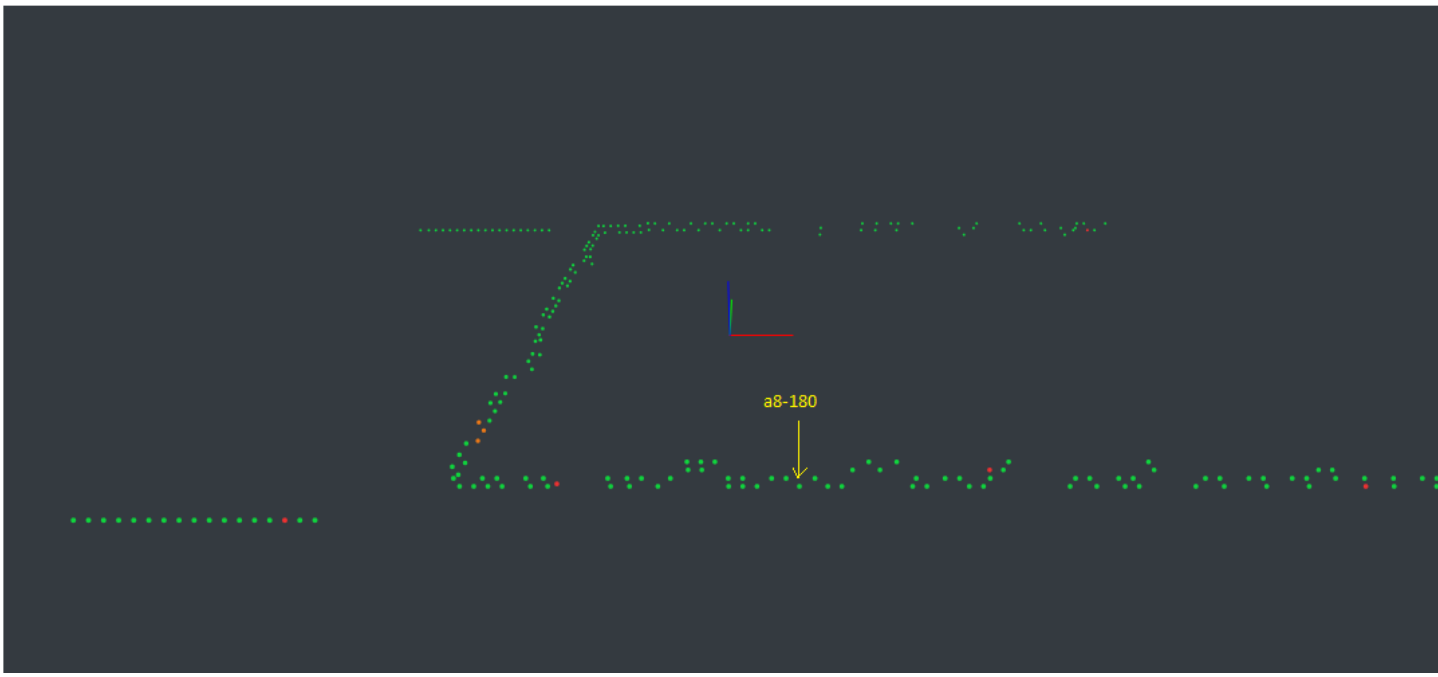
Evaluate the performance of the system as the number of wireless elements increases. How does the physical location of the nodes affect the performance of the wireless network? Examine wireless network topologies of different diameter.

- Measure the energy consumption and duty cycling of the nodes keeping in mind the overall performance of the wireless sensor network.

For the previuos experiments it has been increased the diameter of the wireless network, although maintained the border ruoter a8 node in the same location.

In the fourth and last experiment the border router will be moved centrally in the topology and the overall performance will be measured.

The new topology to be investigated will be as in the following graphic:



---

i

Calibration actions are required every time a sensor is deployed in a different environment, as the physical measurement elements must be adjusted or

even dedicated to the monitored device or process, providing at the start a reduction of measuring uncertainty and minimal interference with sensor functions. However, periodic calibrations are also needed, since during the operation, we can assist the change of conditions with respect to those known during the calibration process and to the impact of various external factors that could be absent in the laboratory calibration conditions.