Tutorial: Contiki-Cooja & FIT IoT-LAB

Presentation · October 2020

DOI: 10.13140/RG.2.2.30701.41442

Alakesh Kalita
Singapore University of Technology and Design
28 PUBLICATIONS 118 CITATIONS

Internet of Things

Tutorial on Contiki OS, Cooja and FIT IoT-LAB

Alakesh Kalita
PhD Scholar
Indian Institute of Technology Guwahati
E-mail: alakesh.kalita025@gmail.com

What is Contiki OS ?

- Contiki is an open source operating system for Internet of Things
  - runs on tiny low-power microcontrollers

- It allows to develop applications that make efficient use of different hardware for IoT
  - while providing standardized low-power wireless communication for a range of hardware platforms
  - Mainly, focus on low-power wireless IoT devices

- The Contiki system includes a sensor simulator called Cooja,
  - Cooja simulates of Contiki nodes

How to install Contiki-NG

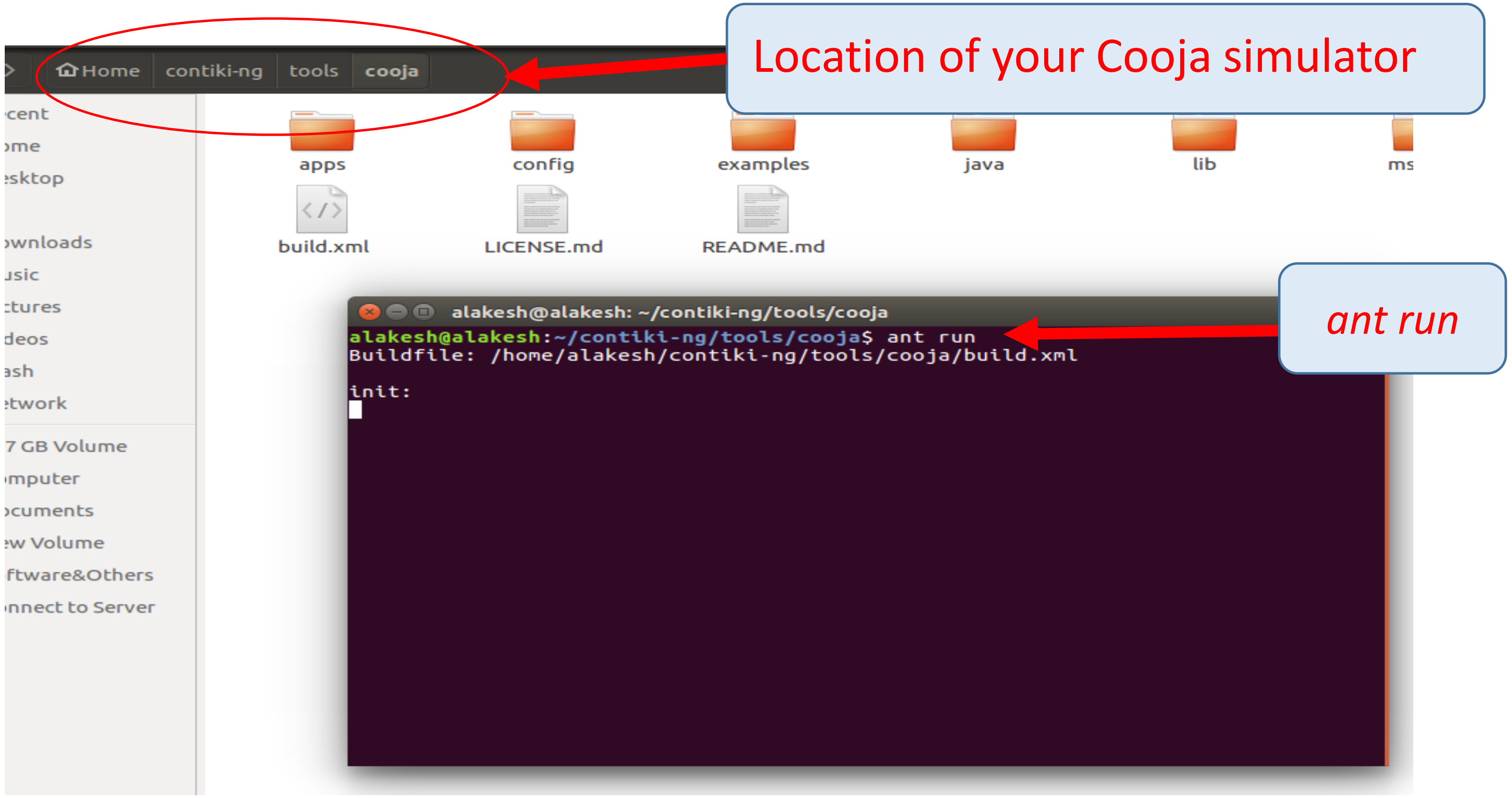
- The latest version of Contiki is known as Contiki-NG (Contiki Next Generation)
- Complete installation procedure of Contiki-NG on Linux can be found at the below link,
  - https://github.com/contiki-ng/contiki-ng/wiki/Toolchain-installation-on-Linux
- One can install Contiki-NG using Virtual Box on Windows OS.

What is Cooja?

- Cooja is a Contiki network simulator
  - To perform IoT network simulations
  - An extensible Java-based simulator capable of emulating various IoT motes such as Tmote sky, Z1 etc.,
  - The code to be executed by the node is the exact same firmware that can be uploaded to physical nodes
  - Allows large and small networks of motes can be simulated at hardware level
- Cooja is a highly useful tool for Contiki development
  - It allows developers to test their code and systems before running it on the target hardware

How to start Cooja?

- After Contiki-NG installation, start Cooja Simulator using the command “*ant run*” inside the Cooja directory of Contiki-NG.



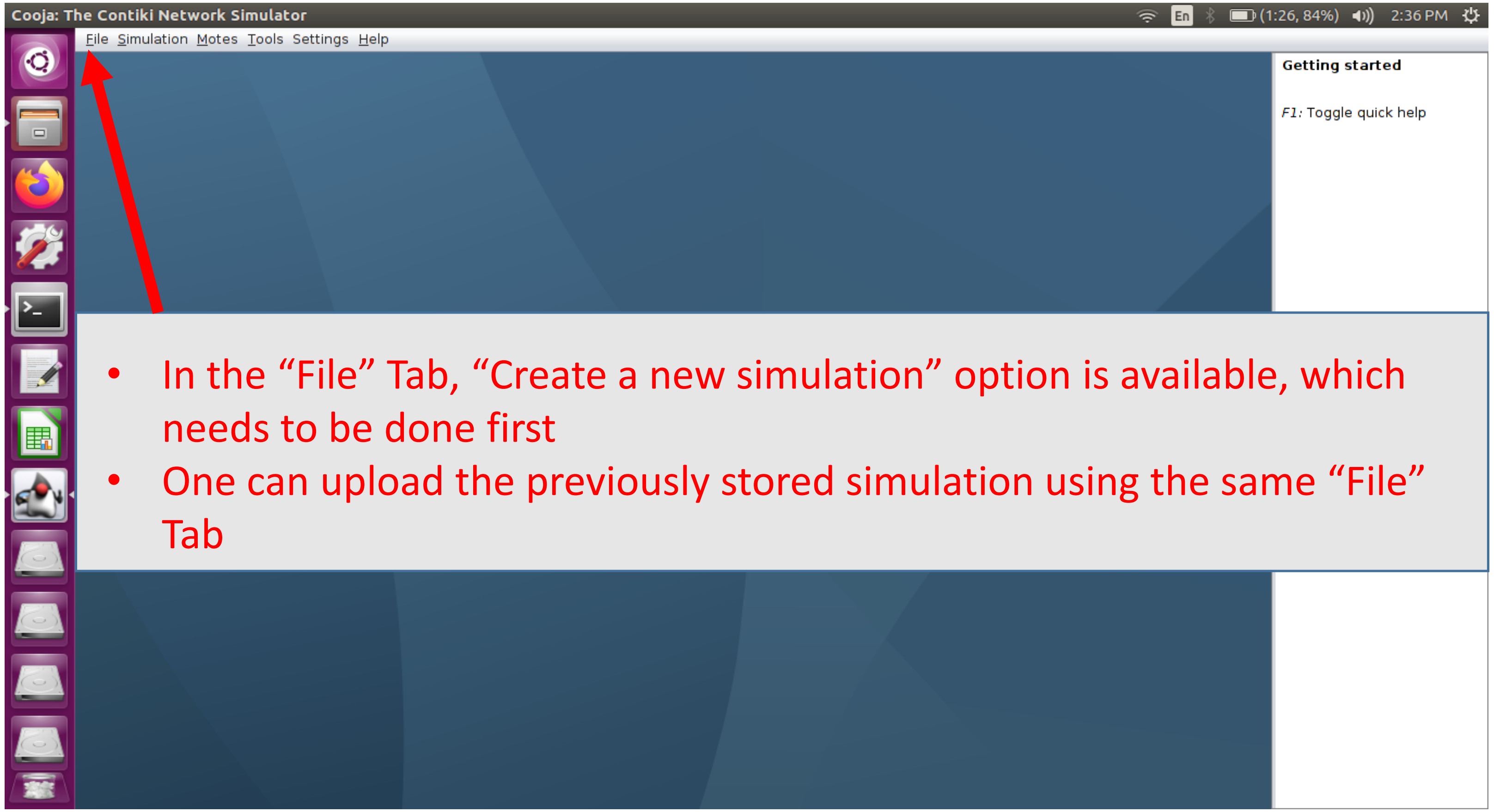
Location of your Cooja simulator

ant run

```
alakesh@alakesh:~/contiki-ng/tools/cooja$ ant run
Buildfile:  /home/alakesh/contiki-ng/tools/cooja/build.xml

init:
```

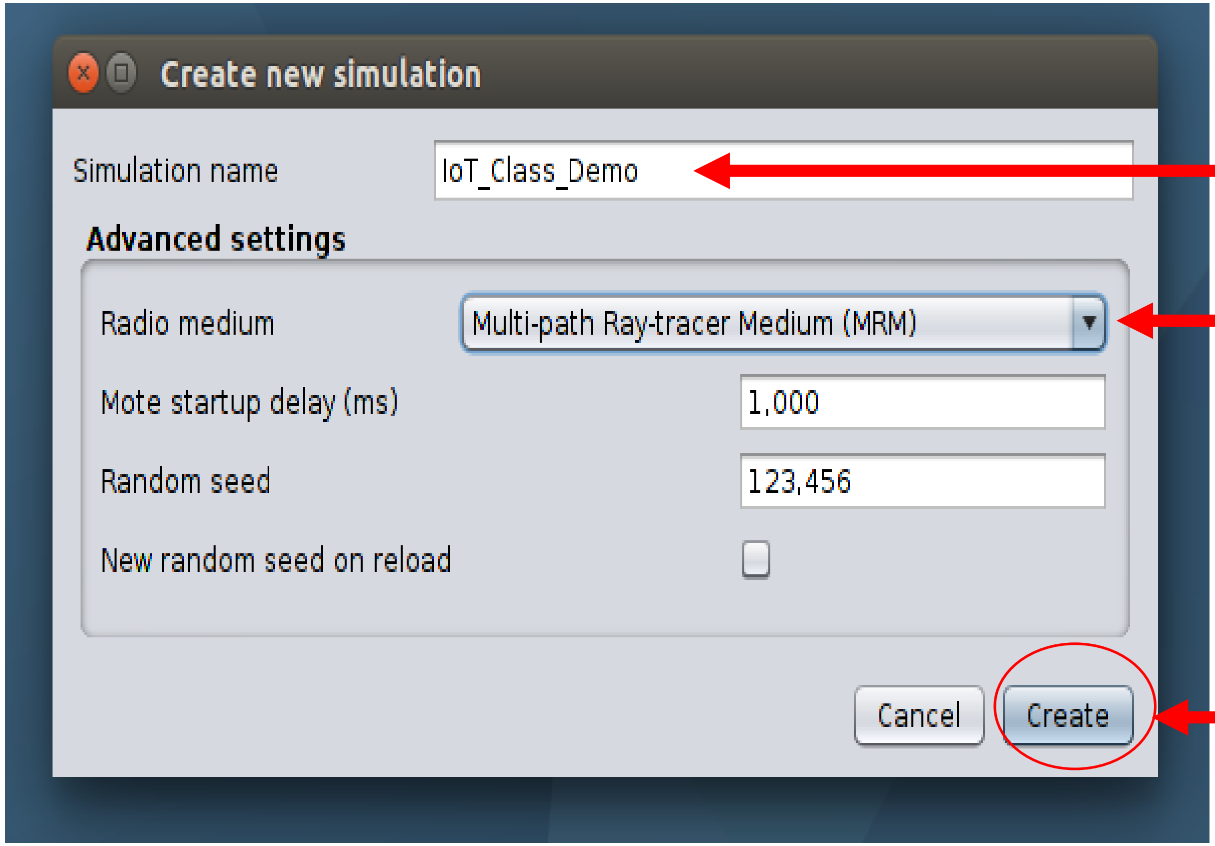
Contd..

➢ First interface after the starting of the simulator


Cooja: The Contiki Network Simulator
(1:26, 84%)
2:36 PM
File Simulation Motes Tools Settings Help
Getting started
F1: Toggle quick help

- In the “File” Tab, “Create a new simulation” option is available, which needs to be done first
- One can upload the previously stored simulation using the same “File” Tab

Create a new Simulation



Create new simulation

Simulation name: IoT_Class_Demo

Advanced settings

Radio medium: Multi-path Ray-tracer Medium (MRM)

Mote startup delay (ms): 1,000

Random seed: 123,456

New random seed on reload: □

Simulation Name

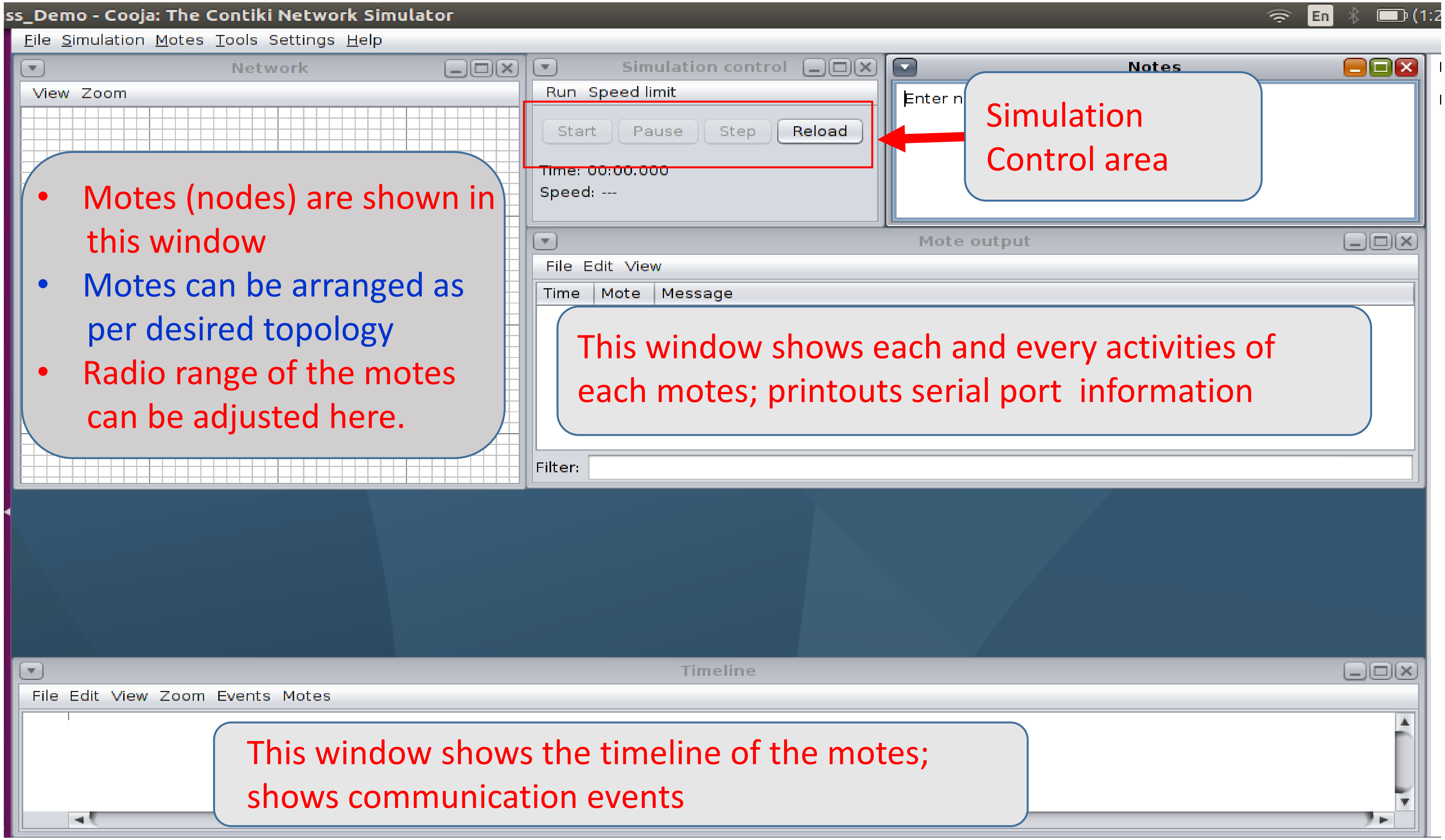
Select radio medium

1. UDGM
2. MRM

MRM gives more realistic environment for simulation

To create the simulation

Basic Simulation Interface



- Motes (nodes) are shown in this window
- Motes can be arranged as per desired topology
- Radio range of the motes can be adjusted here.

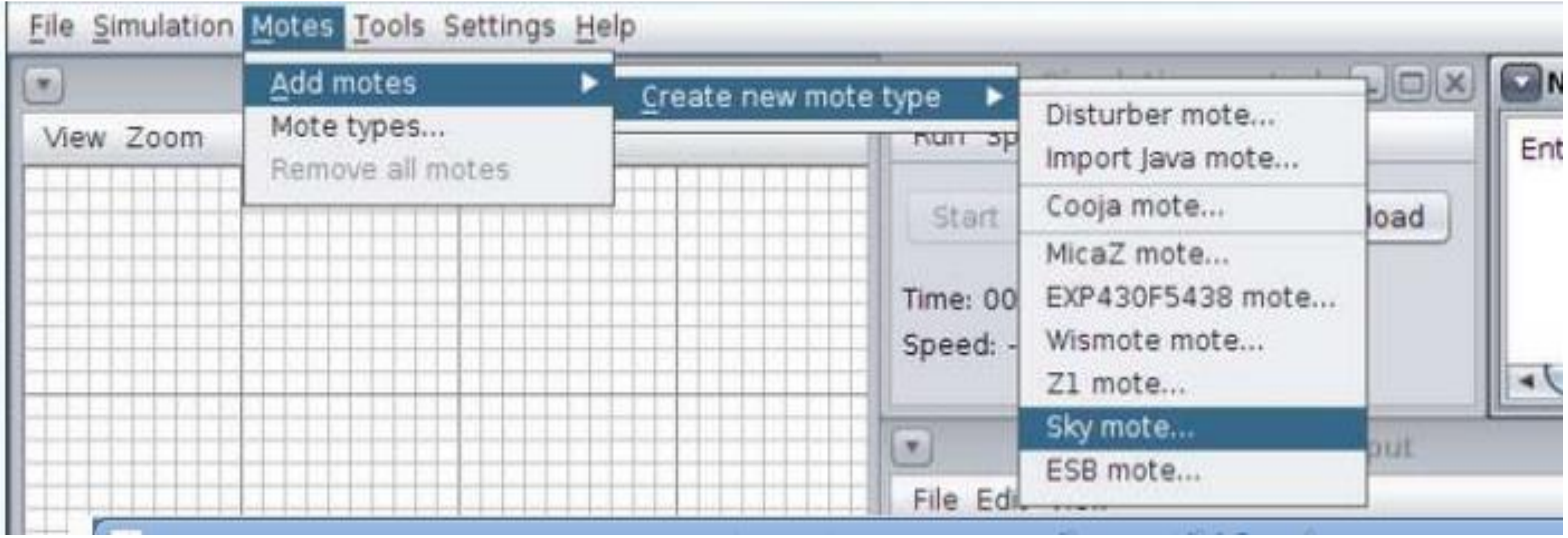
Simulation Control area

This window shows each and every activities of each motes; printouts serial port information

This window shows the timeline of the motes; shows communication events

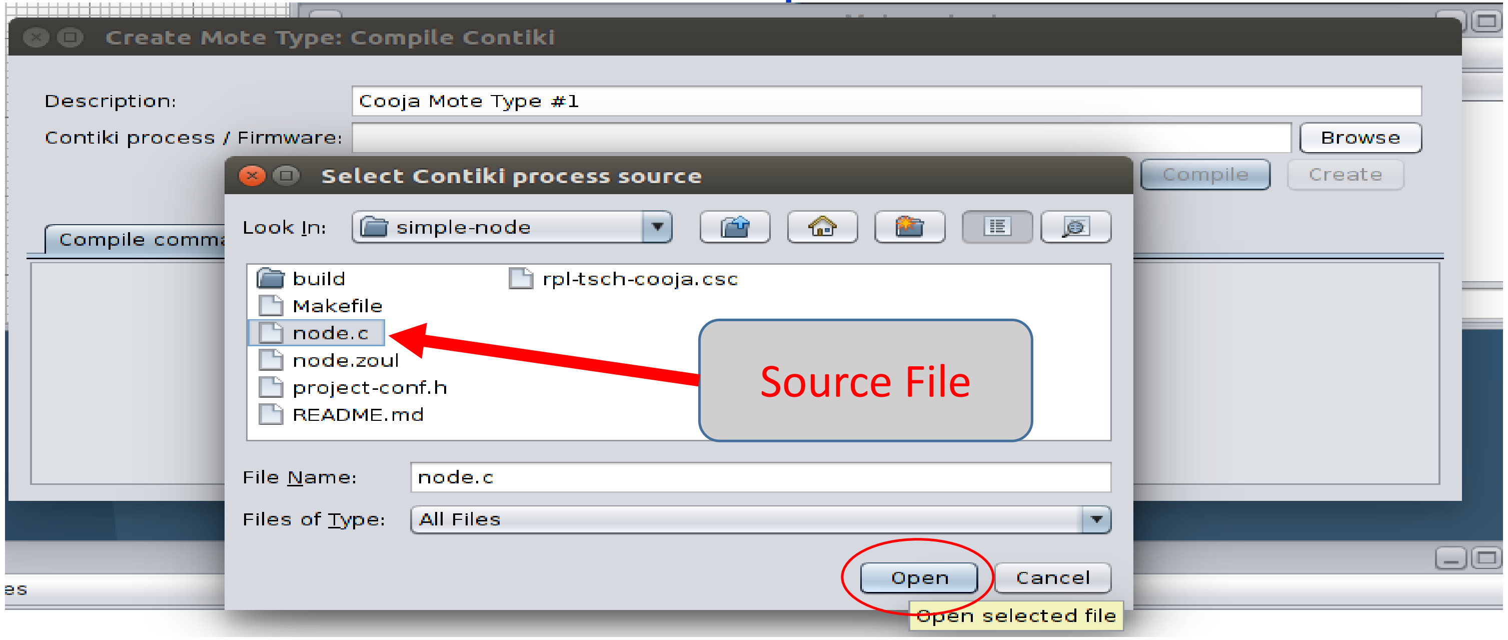
How to add motes (nodes)?

- Before simulation, motes need to be added
- In the “Motes” tab, click on “Add motes”
- Next, click on “Create new mote type” and select the desired available mote (e.g., Cooja mote, sky mote)
- Better to use “Cooja” mote on low configuration system



Select the source file

- Browse the source file that you want to simulate
- In the "example" folder of Contiki-NG, various source files are available
- Select the "node.c" file to compile and then simulate



Create Mote Type: Compile Contiki

Description: Cooja Mote Type #1

Contiki process / Firmware:

Select Contiki process source

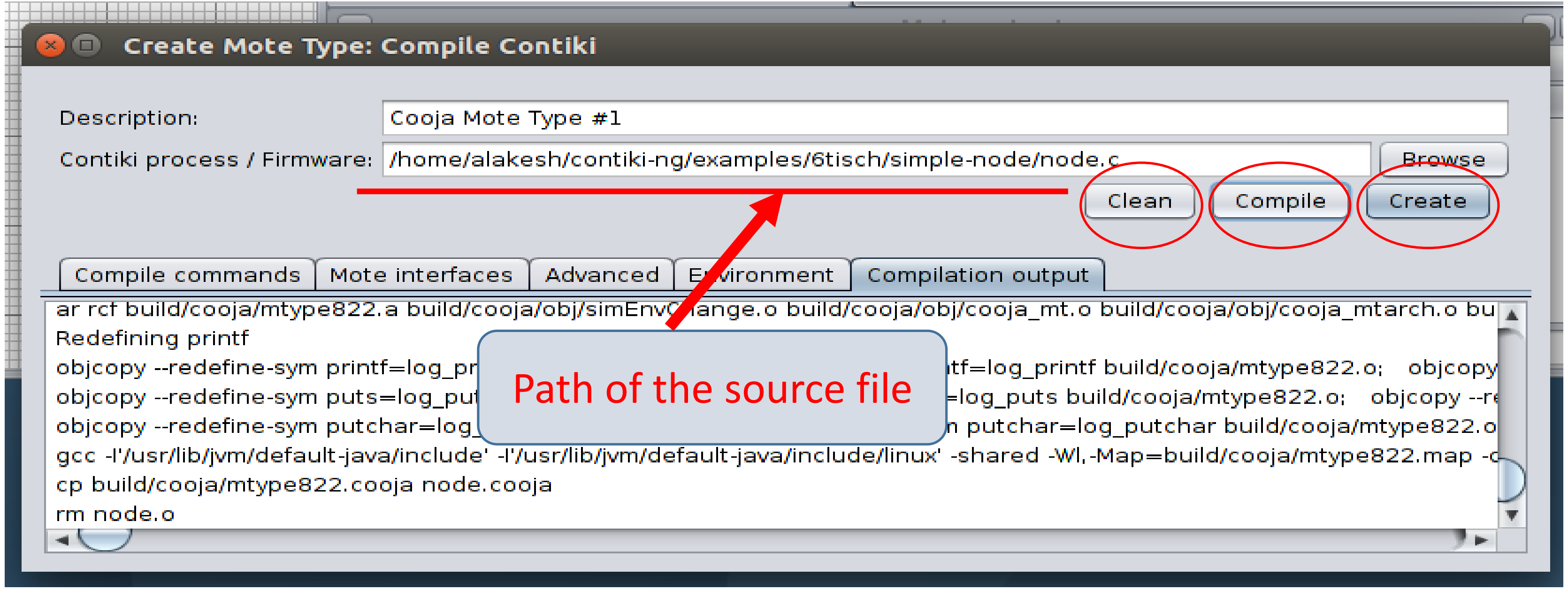
Look In: simple-node

File Name: node.c

Files of Type: All Files

Contd..

- Before Simulation, selected source file needs to be compiled.
- If any changes made in any of the source file, then click on “clean” before compiling, otherwise, no cleaning is required
- Click on “compile” to compile and then “create” to create the motes



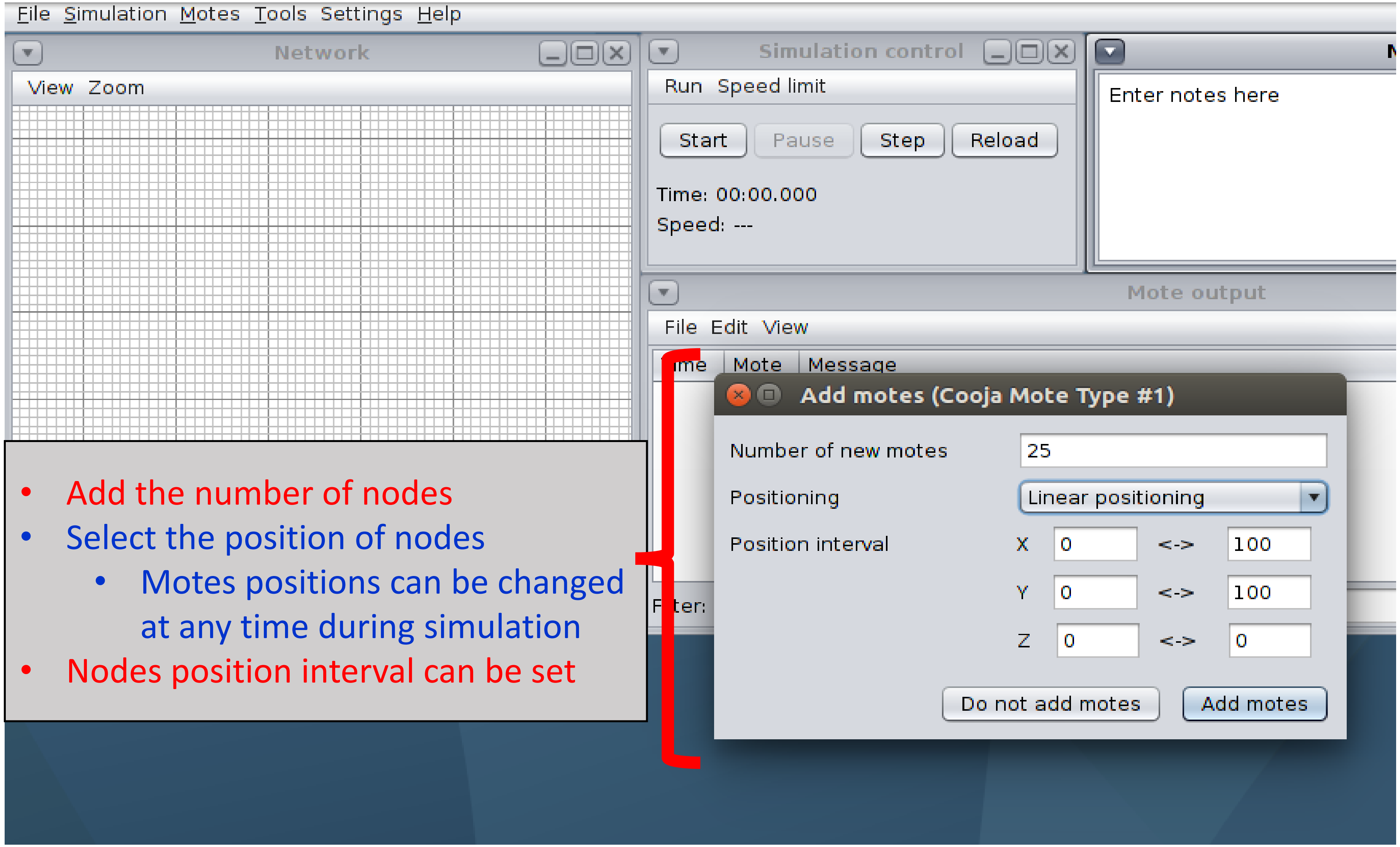
Description: Cooja Mote Type #1

Contiki process / Firmware: /home/alakesh/contiki-ng/examples/6tisch/simple-node/node.c

```
ar rcf build/cooja/mtype822.a build/cooja/obj/simEnvC ange.o build/cooja/obj/cooja_mt.o build/cooja/obj/cooja_mtarch.o bu
Redefining printf
objcopy --redefine-sym printf=log_pr                 tf=log_printf build/cooja/mtype822.o;   objcopy
objcopy --redefine-sym puts=log_put                 =log_puts build/cooja/mtype822.o;   objcopy --re
objcopy --redefine-sym putchar=log_                 n putchar=log_putchar build/cooja/mtype822.o
gcc -I'/usr/lib/jvm/default-java/include' -I'/usr/lib/jvm/default-java/include/linux' -shared -Wl,-Map=build/cooja/mtype822.map -c
cp build/cooja/mtype822.cooja node.cooja
rm node.o
```

Path of the source file

Contd..



Time: 00:00.000

Speed: ---

Add motes (Cooja Mote Type #1)

Number of new motes: 25

Positioning: Linear positioning

Position interval:

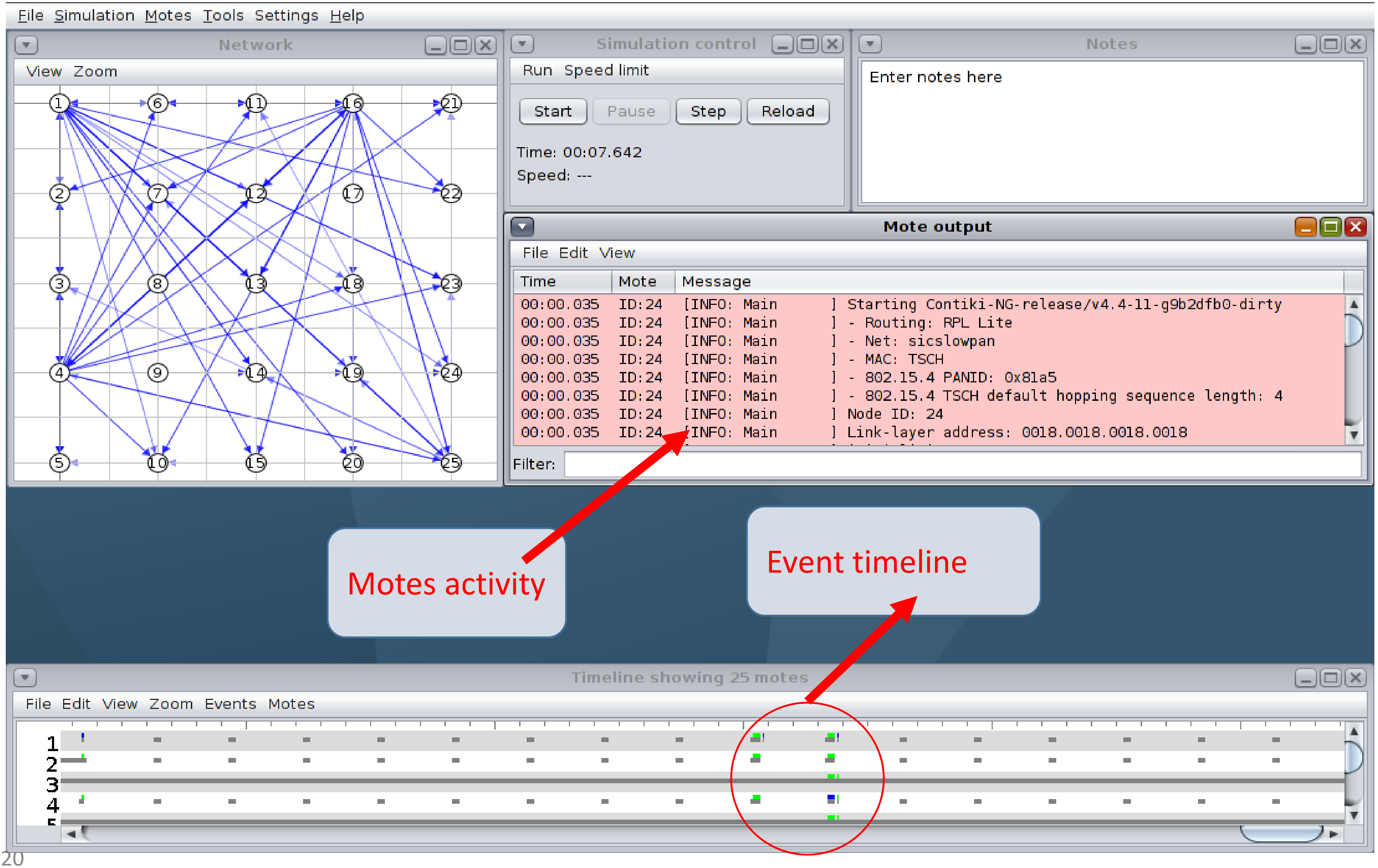
X 0 <-> 100

Y 0 <-> 100

Z 0 <-> 0

- Add the number of nodes
- Select the position of nodes
  - Motes positions can be changed at any time during simulation
- Nodes position interval can be set

Interface during simulation-1



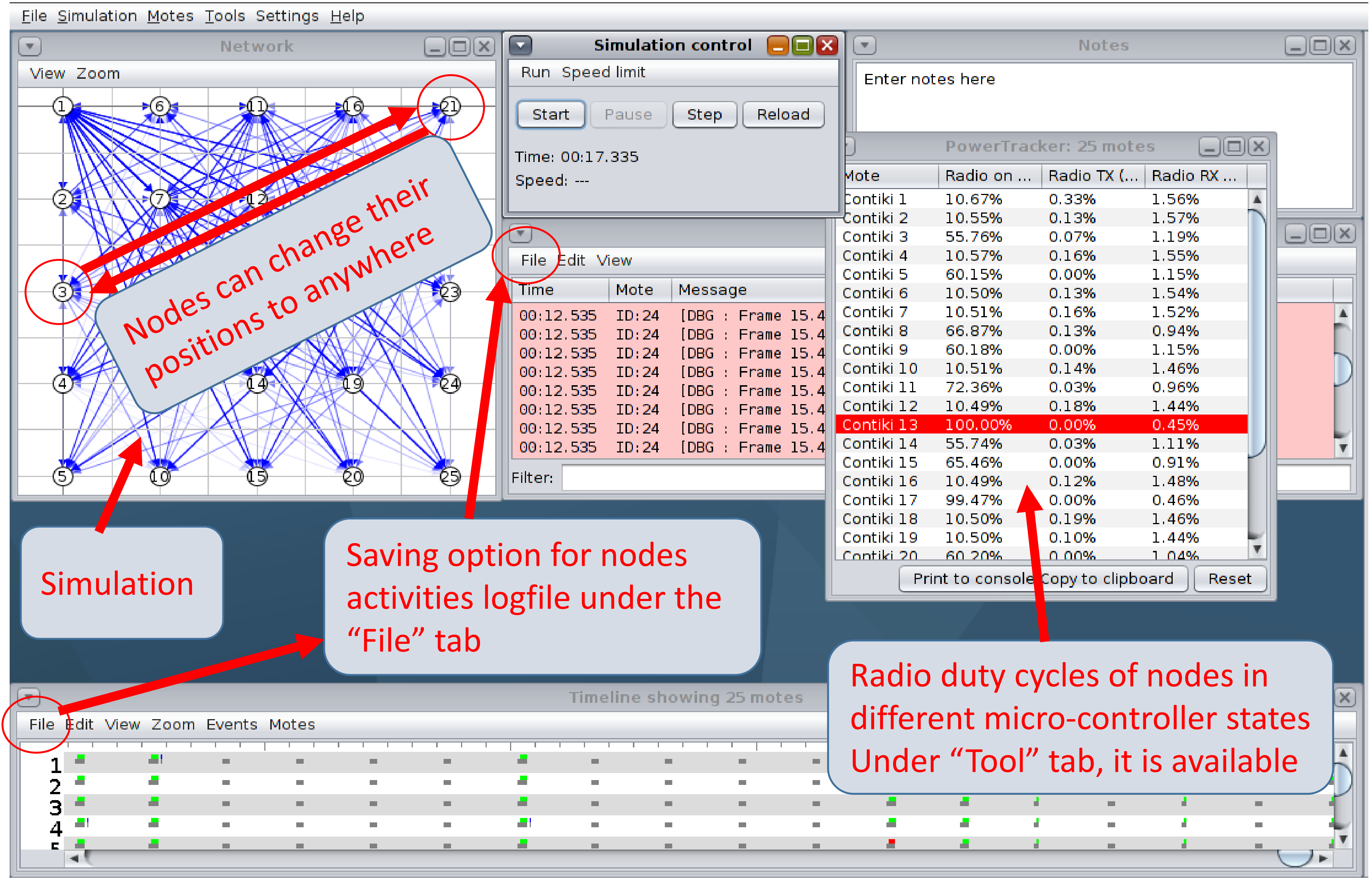
Time: 00:07.642
Speed: ---

```
Time       Mote   Message
00:00.035  ID:24  [INFO: Main      ] Starting Contiki-NG-release/v4.4-11-g9b2dfb0-dirty
00:00.035  ID:24  [INFO: Main      ] - Routing: RPL Lite
00:00.035  ID:24  [INFO: Main      ] - Net: sicslowpan
00:00.035  ID:24  [INFO: Main      ] - MAC: TSCH
00:00.035  ID:24  [INFO: Main      ] - 802.15.4 PANID: 0x81a5
00:00.035  ID:24  [INFO: Main      ] - 802.15.4 TSCH default hopping sequence length: 4
00:00.035  ID:24  [INFO: Main      ] Node ID: 24
00:00.035  ID:24  [INFO: Main      ] Link-layer address: 0018.0018.0018.0018
```

Motes activity

Event timeline

Interface during simulation-2



Time: 00:17.335
Speed: ---

```
Time        Mote    Message
00:12.535   ID:24   [DBG : Frame 15.4
00:12.535   ID:24   [DBG : Frame 15.4
00:12.535   ID:24   [DBG : Frame 15.4
00:12.535   ID:24   [DBG : Frame 15.4
00:12.535   ID:24   [DBG : Frame 15.4
00:12.535   ID:24   [DBG : Frame 15.4
00:12.535   ID:24   [DBG : Frame 15.4
00:12.535   ID:24   [DBG : Frame 15.4
```

PowerTracker: 25 motes

Mote	Radio on ...	Radio TX (...	Radio RX ...
Contiki 1	10.67%	0.33%	1.56%
Contiki 2	10.55%	0.13%	1.57%
Contiki 3	55.76%	0.07%	1.19%
Contiki 4	10.57%	0.16%	1.55%
Contiki 5	60.15%	0.00%	1.15%
Contiki 6	10.50%	0.13%	1.54%
Contiki 7	10.51%	0.16%	1.52%
Contiki 8	66.87%	0.13%	0.94%
Contiki 9	60.18%	0.00%	1.15%
Contiki 10	10.51%	0.14%	1.46%
Contiki 11	72.36%	0.03%	0.96%
Contiki 12	10.49%	0.18%	1.44%
Contiki 13	100.00%	0.00%	0.45%
Contiki 14	55.74%	0.03%	1.11%
Contiki 15	65.46%	0.00%	0.91%
Contiki 16	10.49%	0.12%	1.48%
Contiki 17	99.47%	0.00%	0.46%
Contiki 18	10.50%	0.19%	1.46%
Contiki 19	10.50%	0.10%	1.44%
Contiki 20	60.20%	0.00%	1.04%

Nodes can change their positions to anywhere

Simulation

Saving option for nodes activities logfile under the “File” tab

Radio duty cycles of nodes in different micro-controller states Under “Tool” tab, it is available

FIT IoT-LAB

The Very Large Scale IoT Testbed

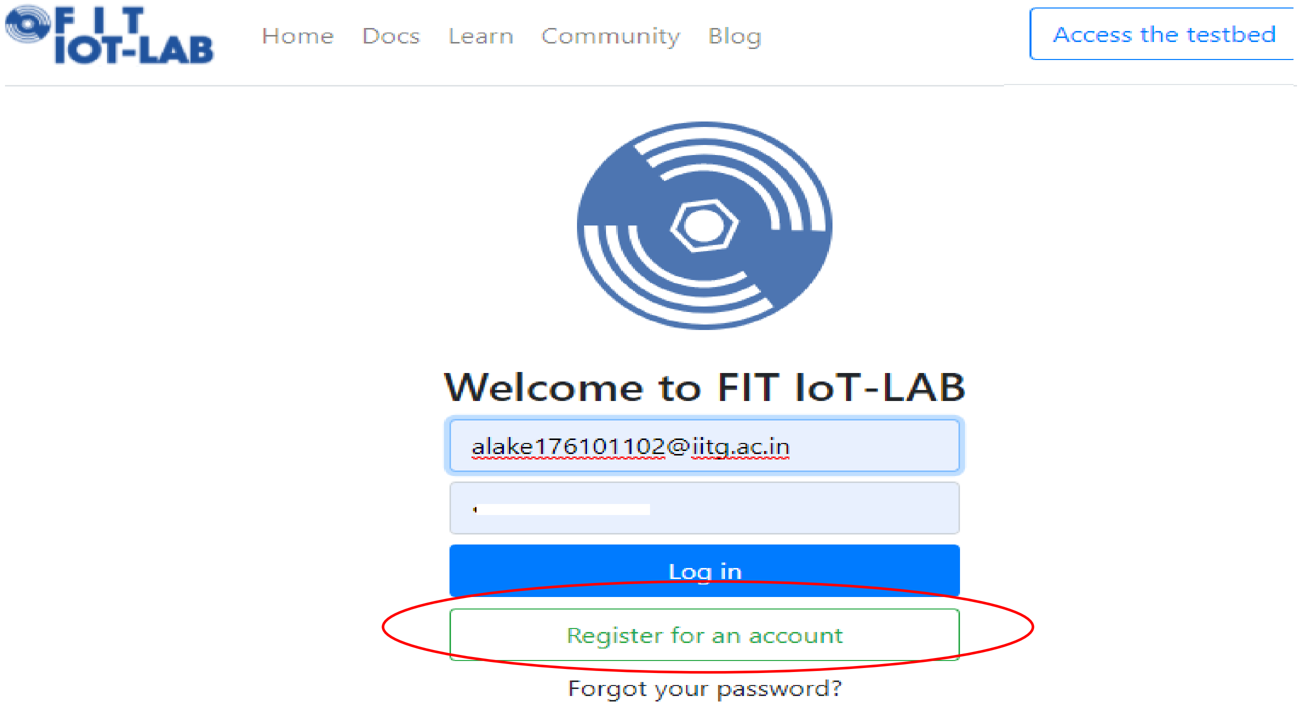
IoT-LAB provides a facility suitable for testing networking with small wireless sensor devices and heterogeneous communicating objects.

What is FIT IoT-LAB?

- FIT IoT-LAB is a real open source IoT Testbed, can be accessible over Internet
  - where, real physical nodes are deployed
  - provides a very large scale infrastructure for testing
    - small wireless sensor devices
    - and heterogeneous communicating objects.
- FIT IoT-LAB features over 1500 wireless sensor nodes spread across six different sites in France.
- Nodes are either fixed or mobile and can be allocated in various topologies throughout all sites
- Node are with different processor architectures (MSP430, STM32 and Cortex-A8) and different wireless chips (802.15.4 PHY @ 800 MHz or 2.4 GHz)

How to access FIT IoT-LAB?

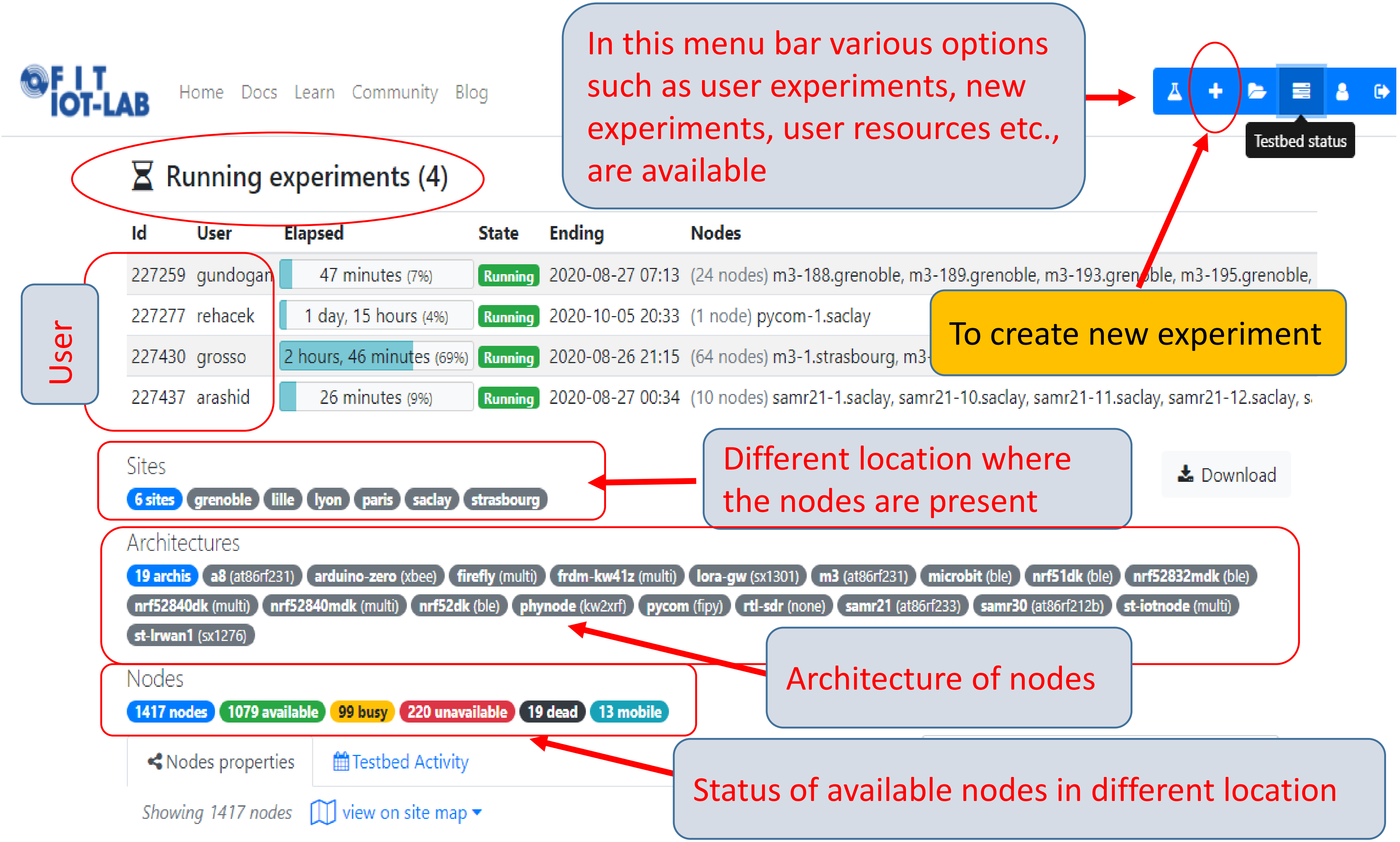
- FIT IoT-LAB is available on https://www.iot-lab.info/
- To access the testbed, one should create his/her account in FIT IoT-LAB



Welcome to FIT IoT-LAB

alake176101102@iitg.ac.in

Testbed Status: after login



In this menu bar various options such as user experiments, new experiments, user resources etc., are available

To create new experiment

User

⌛ Running experiments (4)

Id	User	Elapsed	State	Ending	Nodes
227259	gundogan	47 minutes (7%)	Running	2020-08-27 07:13	(24 nodes) m3-188.grenoble, m3-189.grenoble, m3-193.grenoble, m3-195.grenoble,
227277	rehacek	1 day, 15 hours (4%)	Running	2020-10-05 20:33	(1 node) pycom-1.saclay
227430	grosso	2 hours, 46 minutes (69%)	Running	2020-08-26 21:15	(64 nodes) m3-1.strasbourg, m3-
227437	arashid	26 minutes (9%)	Running	2020-08-27 00:34	(10 nodes) samr21-1.saclay, samr21-10.saclay, samr21-11.saclay, samr21-12.saclay, s

Sites

6 sites | grenoble | lille | lyon | paris | saclay | strasbourg

Different location where the nodes are present

Download

Architectures

19 archis | a8 (at86rf231) | arduino-zero (xbee) | firefly (multi) | frdm-kw41z (multi) | lora-gw (sx1301) | m3 (at86rf231) | microbit (ble) | nrf51dk (ble) | nrf52832mdk (ble) | nrf52840dk (multi) | nrf52840mdk (multi) | nrf52dk (ble) | phynode (kw2xrf) | pycom (fipy) | rtl-sdr (none) | samr21 (at86rf233) | samr30 (at86rf212b) | st-iotnode (multi) | st-lrwan1 (sx1276)

Architecture of nodes

Nodes

1417 nodes | 1079 available | 99 busy | 220 unavailable | 19 dead | 13 mobile

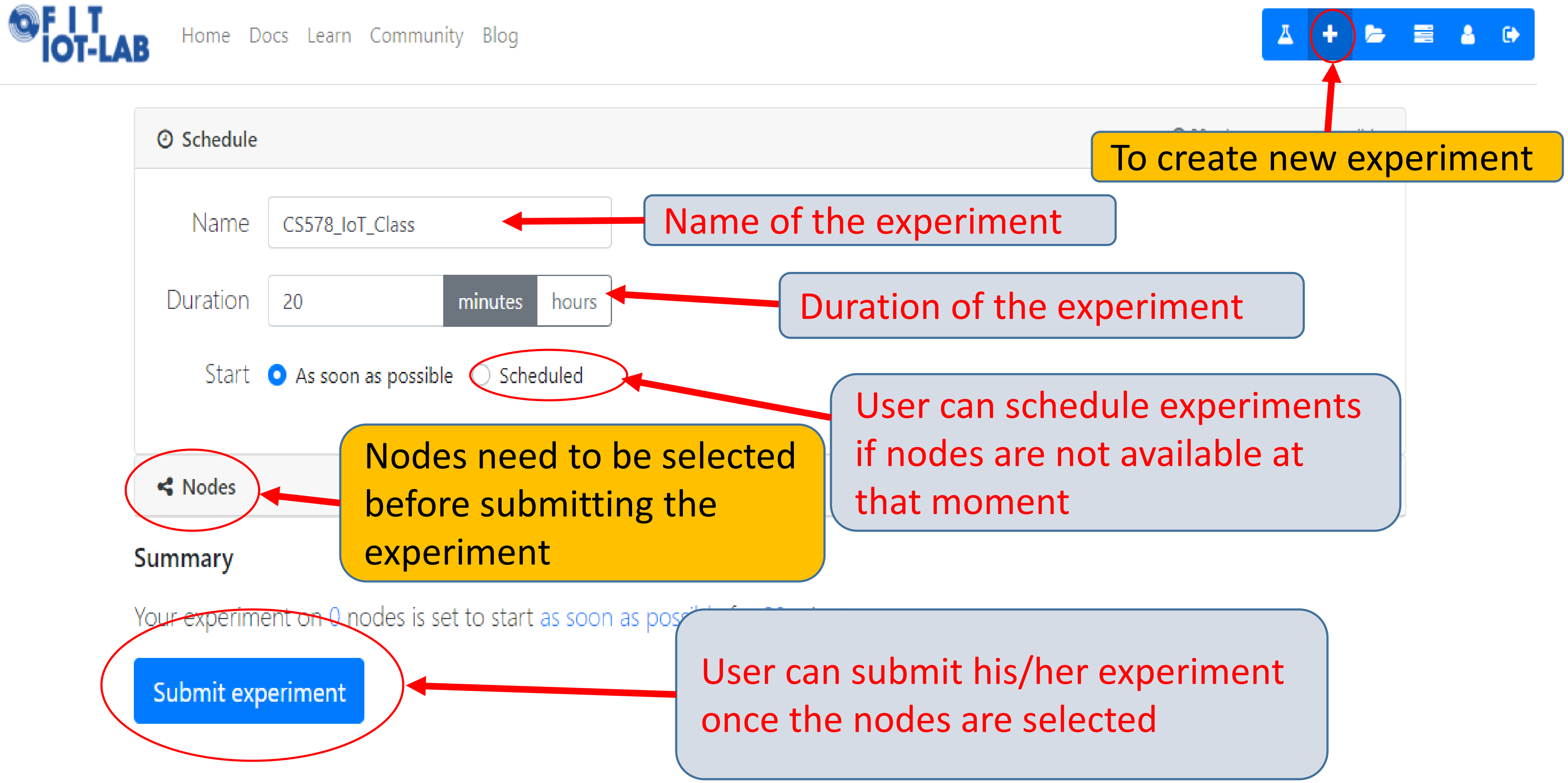
Nodes properties | Testbed Activity

Showing 1417 nodes | view on site map ▾

Status of available nodes in different location

How to perform an experiment?

- To create and perform a new experiment, user needs to create new experiment.



Name: CS578_IoT_Class

Duration: 20 [✓ minutes] hours

Start: ● As soon as possible ○ Scheduled

Summary

Your experiment on 0 nodes is set to start as soon as poss

To create new experiment

Name of the experiment

Duration of the experiment

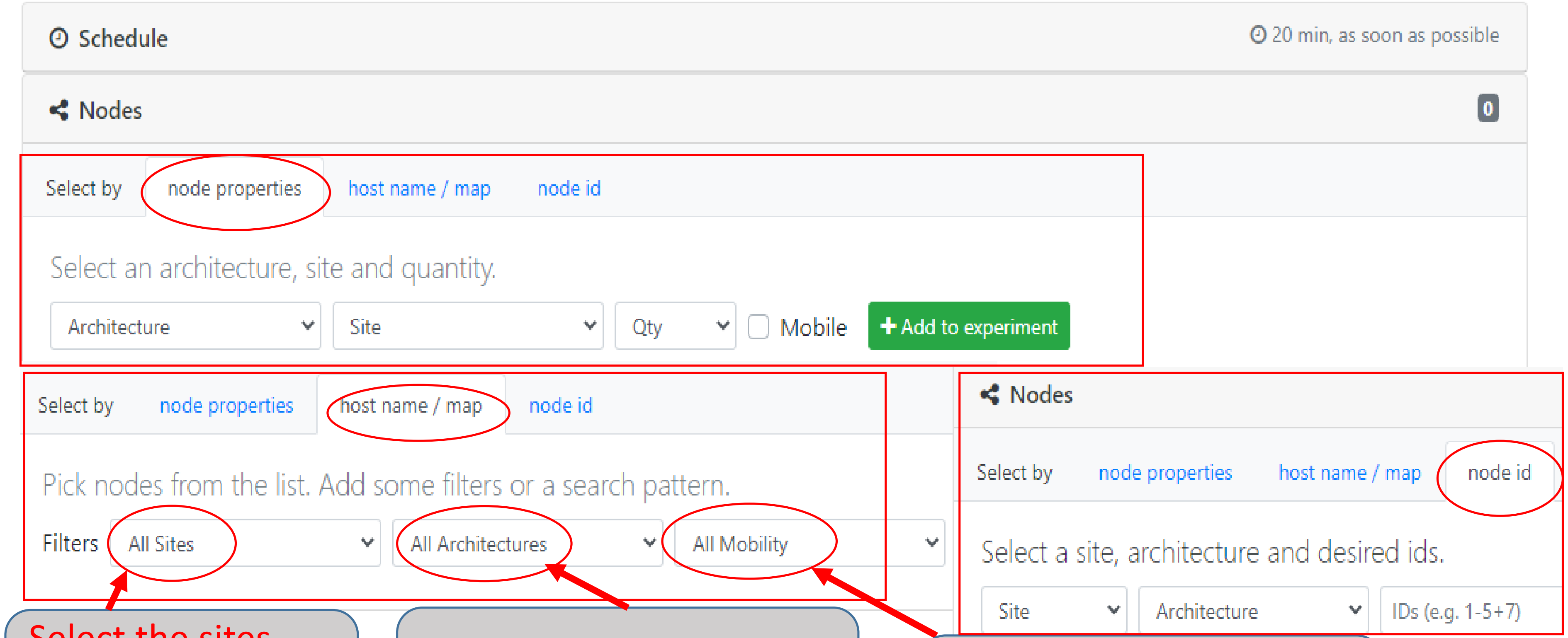
User can schedule experiments if nodes are not available at that moment

Nodes need to be selected before submitting the experiment

User can submit his/her experiment once the nodes are selected

How to select nodes

- User can select nodes by,
  - Node properties, Host name/map and Node Id



Schedule

20 min, as soon as possible

Nodes

0

Select by node properties host name / map node id

Select an architecture, site and quantity.

Architecture | Site | Qty | □ Mobile | + Add to experiment

Select by node properties host name / map node id

Pick nodes from the list. Add some filters or a search pattern.

Filters All Sites | All Architectures | All Mobility

Nodes

Select by node properties host name / map node id

Select a site, architecture and desired ids.

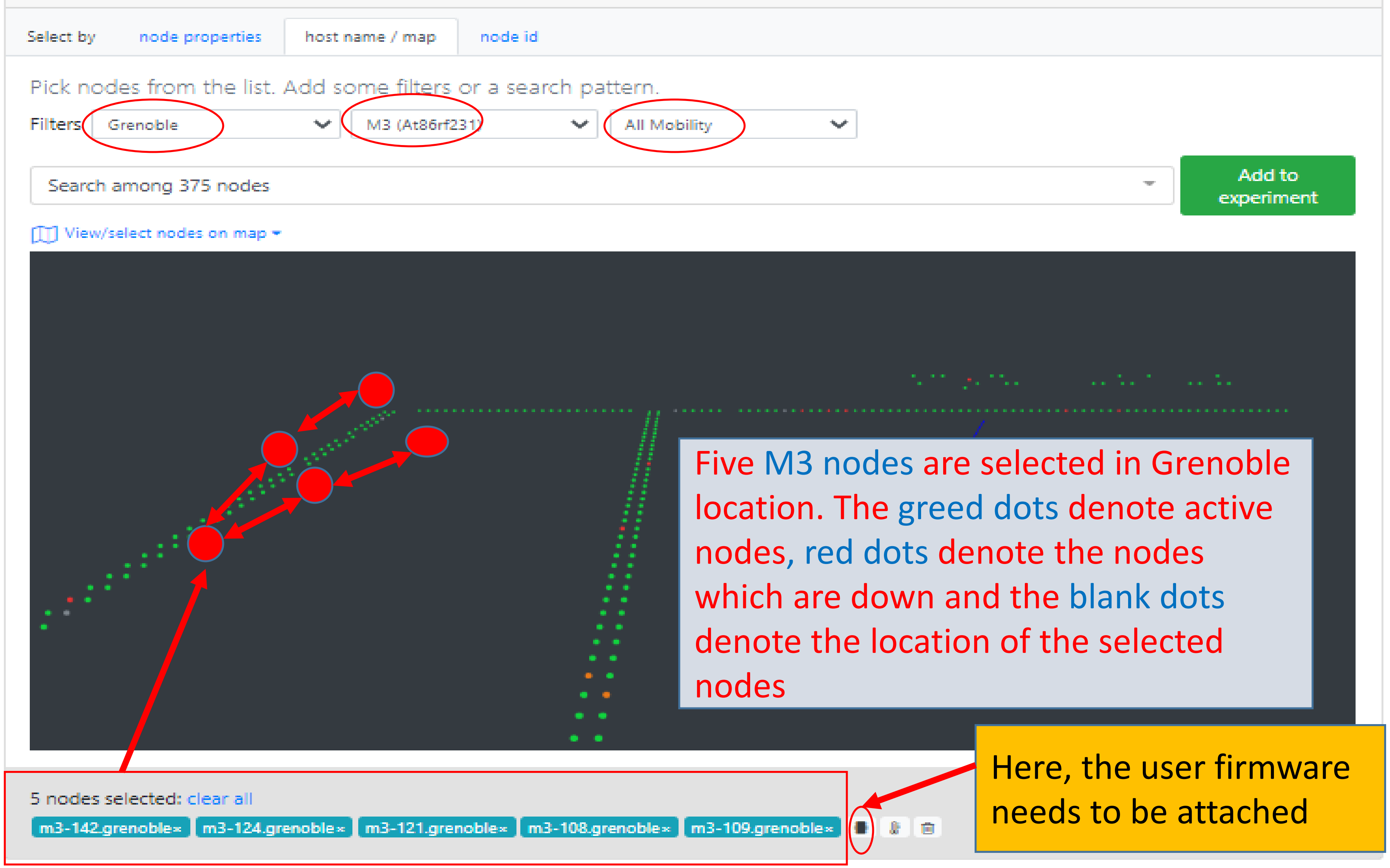
Site | Architecture | IDs (e.g. 1-5+7)

Select the sites such as grenoble, Lille, Lyon, etc.,

Select the architecture such as M3, A8 etc.,

Selects nodes mobilities i.e., mobile or not mobile

Contd..



Select by node properties host name / map node id

Pick nodes from the list. Add some filters or a search pattern.

Filters: Grenoble | M3 (At86rf231) | All Mobility

Search among 375 nodes

5 nodes selected: clear all

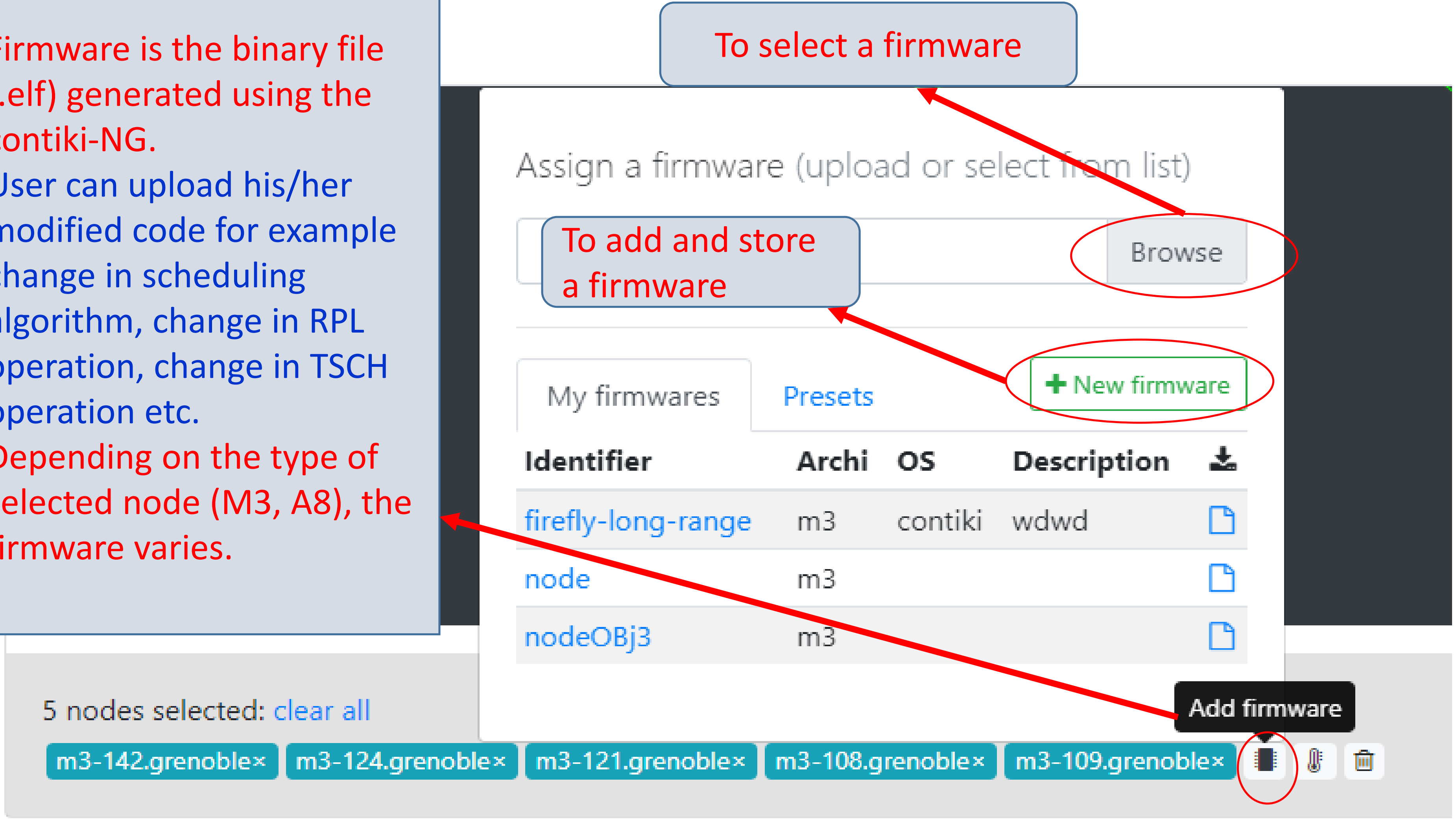
m3-142.grenoble× m3-124.grenoble× m3-121.grenoble× m3-108.grenoble× m3-109.grenoble×

Five M3 nodes are selected in Grenoble location. The greed dots denote active nodes, red dots denote the nodes which are down and the blank dots denote the location of the selected nodes

Here, the user firmware needs to be attached

Contd..

- Firmware is the binary file (.elf) generated using the contiki-NG.
- User can upload his/her modified code for example change in scheduling algorithm, change in RPL operation, change in TSCH operation etc.
- Depending on the type of selected node (M3, A8), the firmware varies.



To select a firmware

To add and store a firmware

Assign a firmware (upload or select from list)

Identifier	Archi	OS	Description	
firefly-long-range	m3	contiki	wdwd	🗋
node	m3			🗋
nodeOBj3	m3			🗋

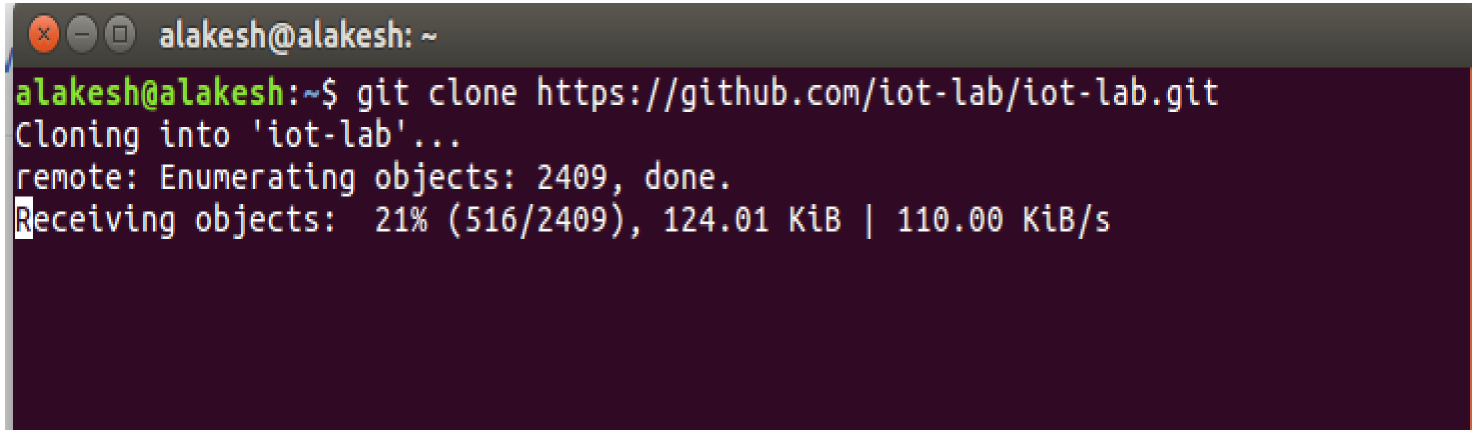
5 nodes selected: clear all

How to generate a firmware

- To compile and deploy firmware, user can directly access the embedded boards by establishing SSH connection.
- The procedure to establish SSH connection can be found in
  - https://www.iot-lab.info/docs/getting-started/ssh-access/
- After establishing the SSH connection, following procedure needs to be followed for firmware compilation,
  - https://www.iot-lab.info/legacy/tutorials/contiki-compilation/index.html

Contd..

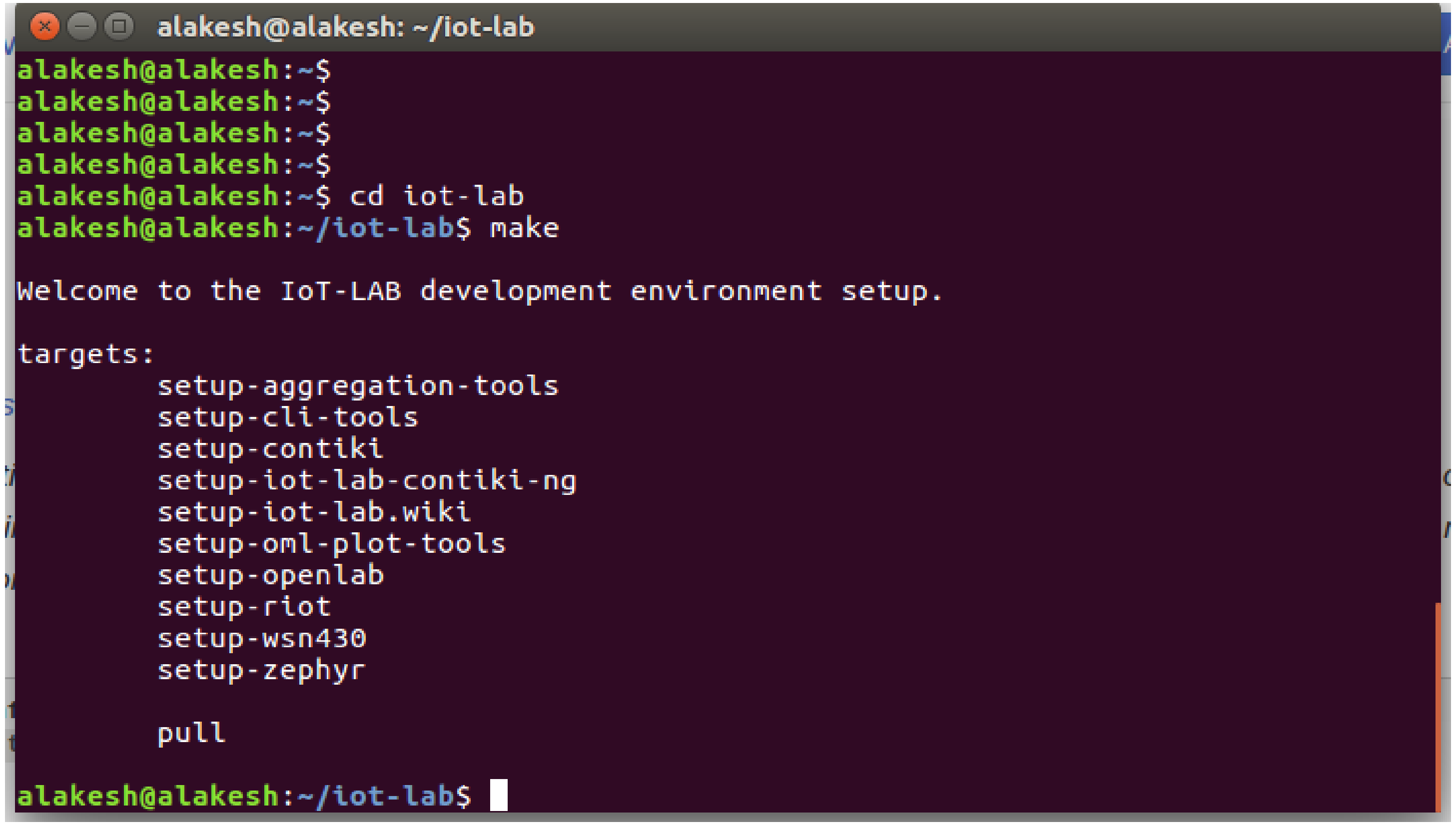
- However, it is better to install the Contiki-NG having the source code for the embedded boards that are available in FIT IoT-LAB on user's own system.
- Steps are as follows,
  - git clone https://github.com/iot-lab/iot-lab.git



```
alakesh@alakesh:~$ git clone https://github.com/iot-lab/iot-lab.git
Cloning into 'iot-lab'...
remote: Enumerating objects: 2409, done.
Receiving objects:  21% (516/2409), 124.01 KiB | 110.00 KiB/s
```

Contd..

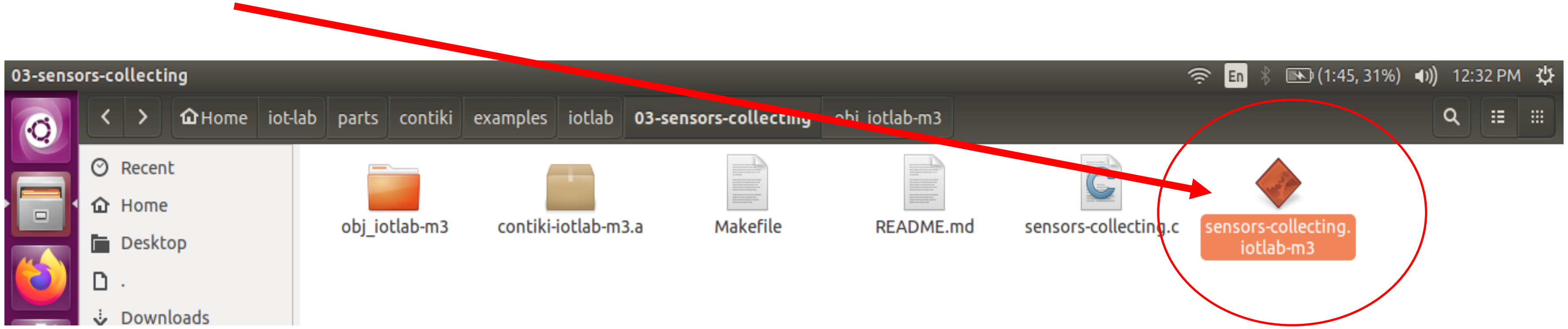
- ➢ *cd iot-lab*
- ➢ *make*



```
alakesh@alakesh:~$
alakesh@alakesh:~$
alakesh@alakesh:~$
alakesh@alakesh:~$
alakesh@alakesh:~$ cd iot-lab
alakesh@alakesh:~/iot-lab$ make

Welcome to the IoT-LAB development environment setup.

targets:
        setup-aggregation-tools
        setup-cli-tools
        setup-contiki
        setup-iot-lab-contiki-ng
        setup-iot-lab.wiki
        setup-oml-plot-tools
        setup-openlab
        setup-riot
        setup-wsn430
        setup-zephyr

        pull

alakesh@alakesh:~/iot-lab$ 
```

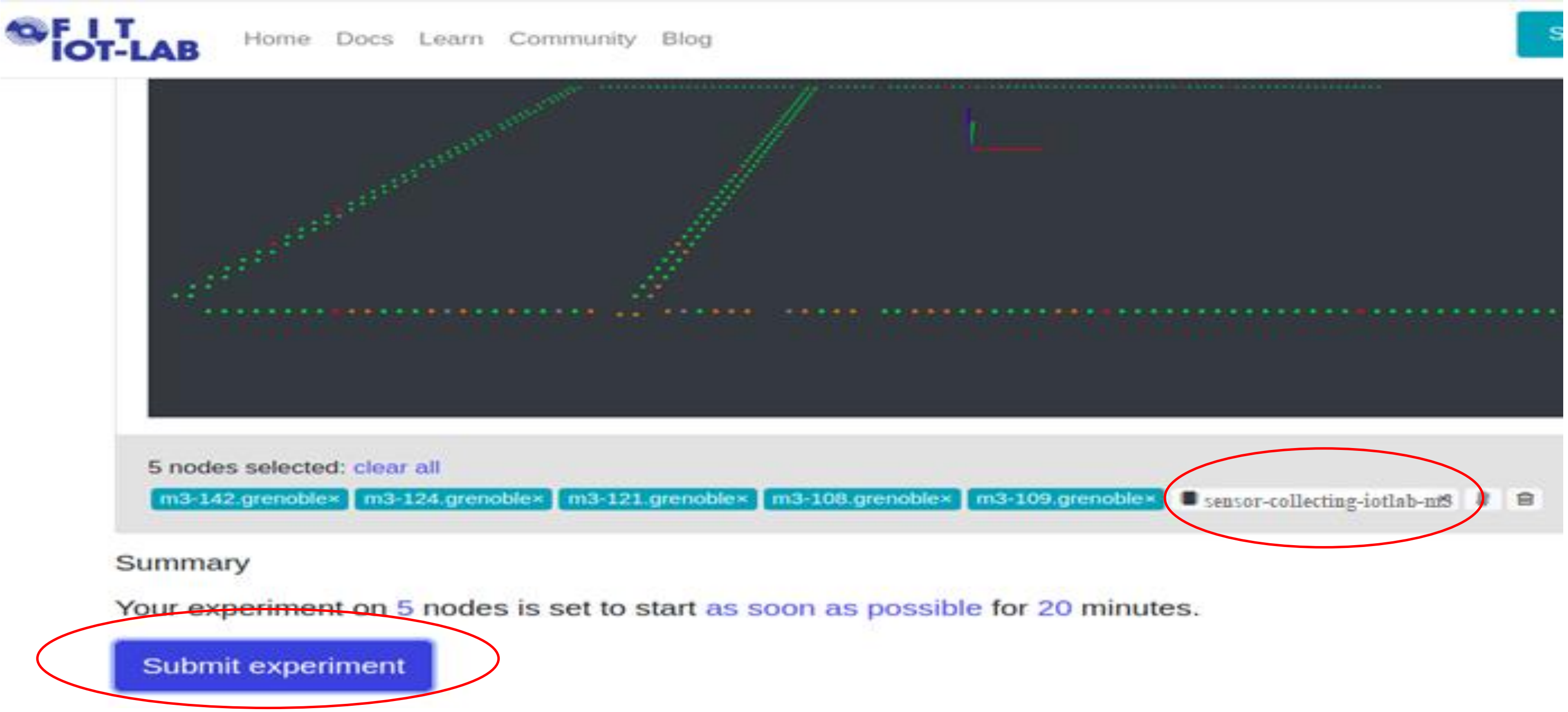
Contd..

- *make setup- contiki*
- Go to the directory
  - *cd parts/ contiki /examples/ iotlab /03-sensors-collecting*
- *make TARGET=iotlab-m3* for m3 nodes and *make TARGET=iotlab-a8-m3* for a8 nodes
- One .elf binary file will be generated. This firmware is needed in FIT IoT-LAB



Upload a firmware

- ➢ Upload the firmware in FIT IoT-LAB
- ➢ After uploading, click on “Submit experiment” bottom to submit your experiment



5 nodes selected: clear all

m3-142.grenoble× m3-124.grenoble× m3-121.grenoble× m3-108.grenoble× m3-109.grenoble×

sensor-collecting-iotlab-m3

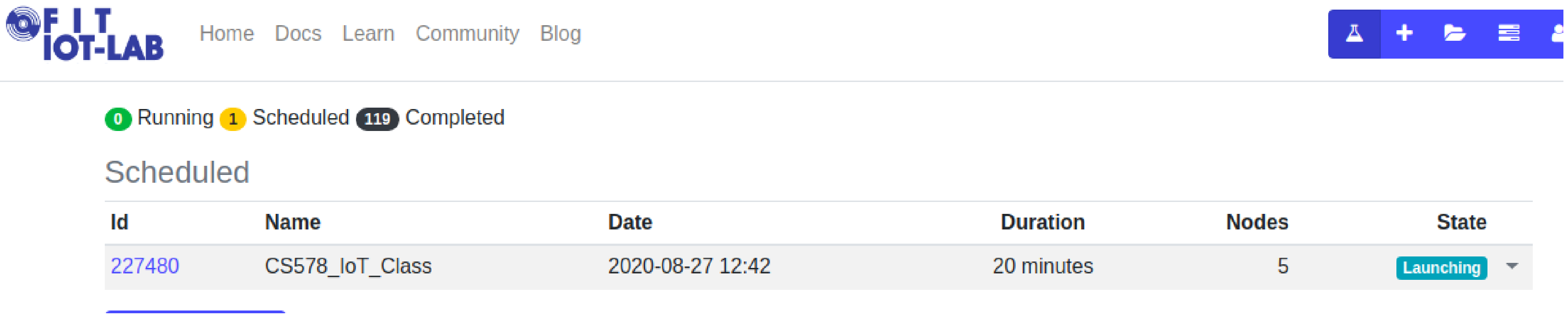
Summary

Your experiment on 5 nodes is set to start as soon as possible for 20 minutes.

Submit experiment

Contd..

- Experiment is launching.

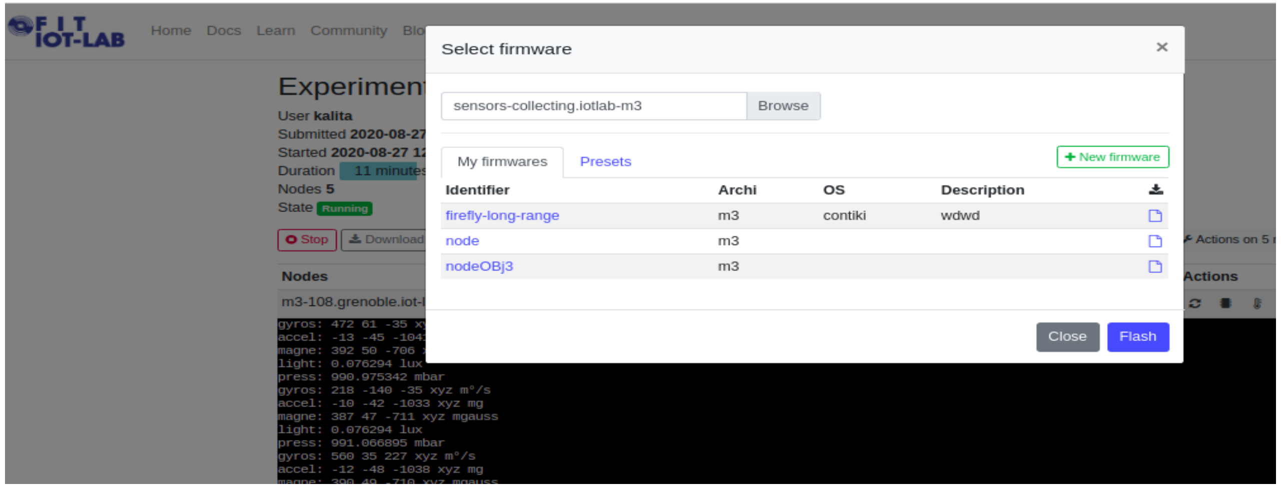


0 Running 1 Scheduled 119 Completed

Scheduled

Id	Name	Date	Duration	Nodes	State
227480	CS578_IoT_Class	2020-08-27 12:42	20 minutes	5	Launching

- Firmware can be updated during experiment also.



Experiment

User **kalita**

Submitted **2020-08-27**

Started **2020-08-27 12**

Duration 11 minutes

Nodes **5**

State Running

Nodes

m3-108.grenoble.iot-l

Actions

```
gyros: 472 61 -35 xy
accel: -13 -45 -104
magne: 392 50 -706
light: 0.076294 lux
press: 990.975342 mbar
gyros: 218 -140 -35 xyz m°/s
accel: -10 -42 -1033 xyz mg
magne: 387 47 -711 xyz mgauss
light: 0.076294 lux
press: 991.066895 mbar
gyros: 560 35 227 xyz m°/s
accel: -12 -48 -1038 xyz mg
magne: 390 49 -710 xyz mgauss
```

Select firmware

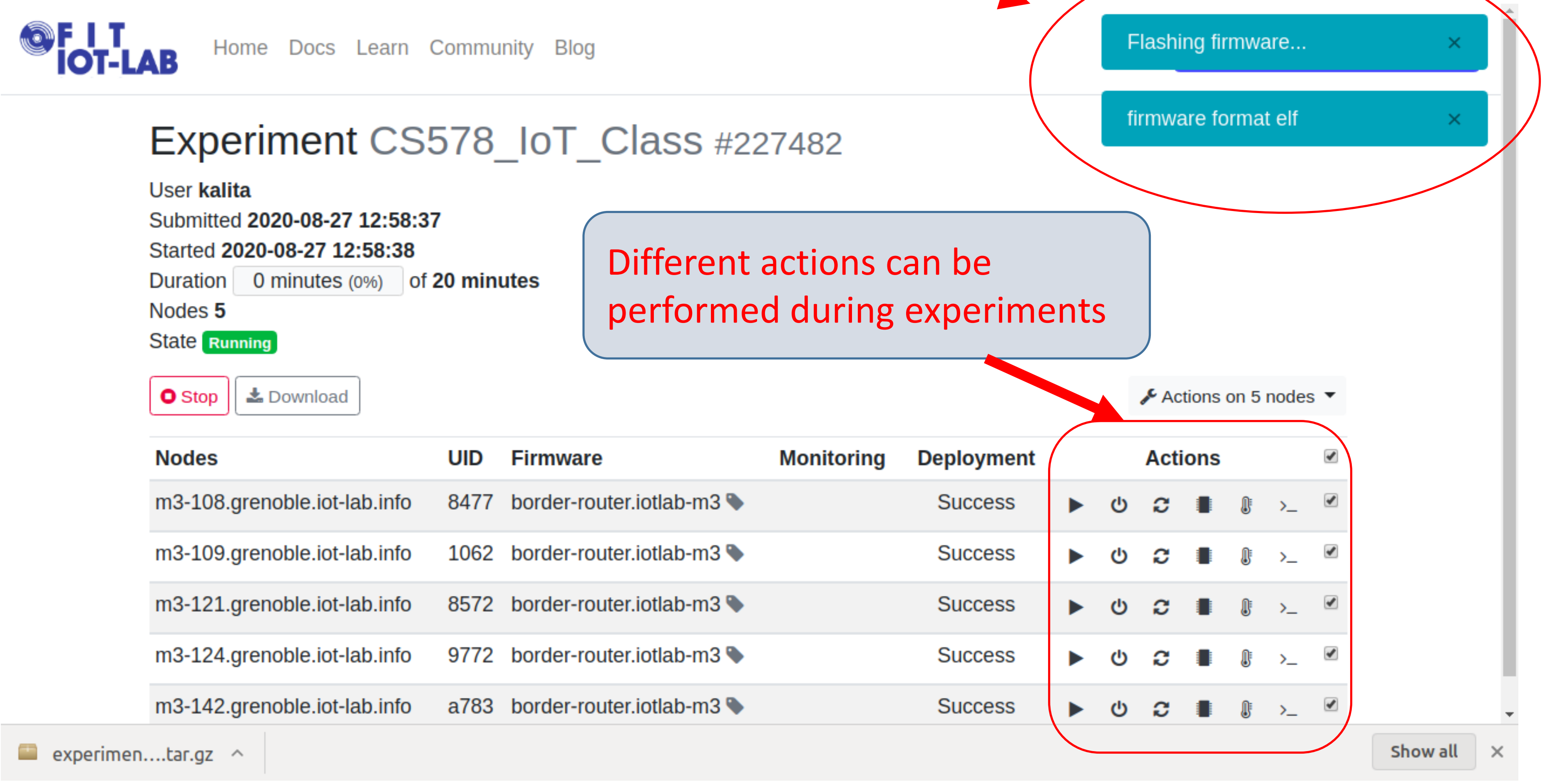
sensors-collecting.iotlab-m3 Browse

My firmwares | Presets | + New firmware

Identifier	Archi	OS	Description
firefly-long-range	m3	contiki	wdwd
node	m3		
nodeOBj3	m3		

Flashing firmware

- ➢ Flashing a firmware during experiment



Home Docs Learn Community Blog

Flashing firmware... ×

firmware format elf ×

Experiment CS578_IoT_Class #227482

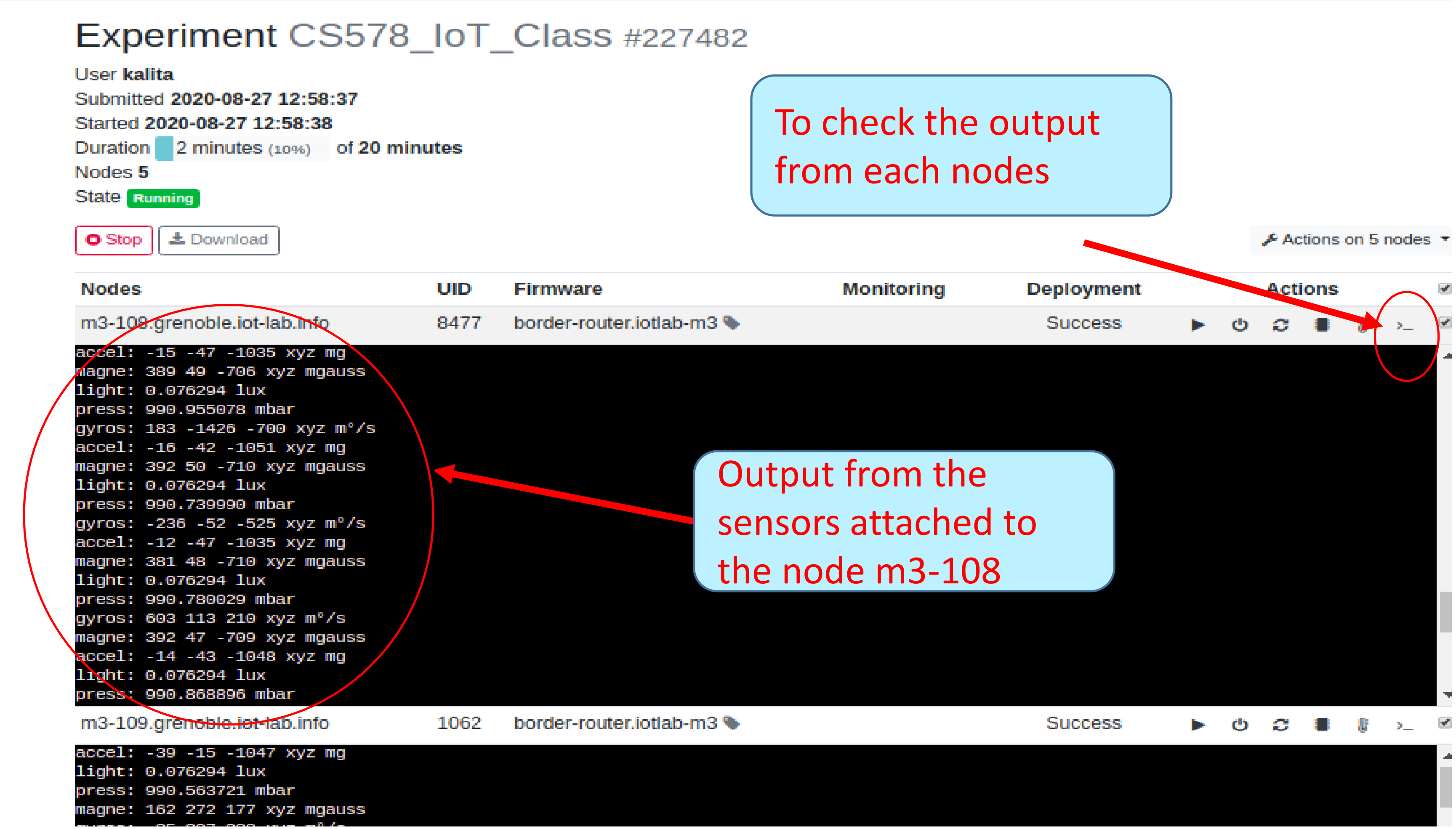
User **kalita**
Submitted **2020-08-27 12:58:37**
Started **2020-08-27 12:58:38**
Duration 0 minutes (0%) of **20 minutes**
Nodes **5**
State **Running**

Different actions can be performed during experiments

Nodes	UID	Firmware	Monitoring	Deployment	Actions	☑
m3-108.grenoble.iot-lab.info	8477	border-router.iotlab-m3		Success	▶ ⏻ ⟳ ▮ 🌡 >_	☑
m3-109.grenoble.iot-lab.info	1062	border-router.iotlab-m3		Success	▶ ⏻ ⟳ ▮ 🌡 >_	☑
m3-121.grenoble.iot-lab.info	8572	border-router.iotlab-m3		Success	▶ ⏻ ⟳ ▮ 🌡 >_	☑
m3-124.grenoble.iot-lab.info	9772	border-router.iotlab-m3		Success	▶ ⏻ ⟳ ▮ 🌡 >_	☑
m3-142.grenoble.iot-lab.info	a783	border-router.iotlab-m3		Success	▶ ⏻ ⟳ ▮ 🌡 >_	☑

Getting the output

- Output can be obtained from each nodes.



Experiment CS578_IoT_Class #227482

User **kalita**
Submitted **2020-08-27 12:58:37**
Started **2020-08-27 12:58:38**
Duration 2 minutes (10%) of **20 minutes**
Nodes **5**
State Running

To check the output from each nodes

Nodes	UID	Firmware	Monitoring	Deployment	Actions
m3-108.grenoble.iot-lab.info	8477	border-router.iotlab-m3		Success	

```
accel: -15 -47 -1035 xyz mg
magne: 389 49 -706 xyz mgauss
light: 0.076294 lux
press: 990.955078 mbar
gyros: 183 -1426 -700 xyz m°/s
accel: -16 -42 -1051 xyz mg
magne: 392 50 -710 xyz mgauss
light: 0.076294 lux
press: 990.739990 mbar
gyros: -236 -52 -525 xyz m°/s
accel: -12 -47 -1035 xyz mg
magne: 381 48 -710 xyz mgauss
light: 0.076294 lux
press: 990.780029 mbar
gyros: 603 113 210 xyz m°/s
magne: 392 47 -709 xyz mgauss
accel: -14 -43 -1048 xyz mg
light: 0.076294 lux
press: 990.868896 mbar
```

Output from the sensors attached to the node m3-108

Nodes	UID	Firmware	Monitoring	Deployment	Actions
m3-109.grenoble.iot-lab.info	1062	border-router.iotlab-m3		Success	

```
accel: -39 -15 -1047 xyz mg
light: 0.076294 lux
press: 990.563721 mbar
magne: 162 272 177 xyz mgauss
```

Thank You

For any query
contact: alakesh.kalita1025@gmail.com