IoT Security – Autumn 2024

Lab 1: Introduction to Contiki - A tiny OS for the Internet of Things

Manh Bui
School of Electrical and Data Engineering
Email: DucManh.Bui@uts.edu.au

About me

- Name: Manh Bui
- You can call me **Manh**
- Research interests: Cybersecurity, Blockchain and Machine Learning
- Email/Teams: DucManh.Bui@uts.edu.au

Aims of Lab 1

- Getting to know **Contiki** and **Cooja Simulator**
- Setting up the lab environment
- Creating the first simulation with Cooja
- Doing some example
- Exercises

What is Contiki?

- An **open-source** operating system for the **Internet of Things**
- Connects tiny **low-cost**, **low-power** microcontrollers to the Internet
- A powerful toolbox for building complex wireless systems
- Support a variety of hardware
- Open source, contributors from: SICS, Cisco, Redwire LLC, and many others.

Arduino

Arduino Pro Mini 328

Raspberry Pi

Particle Photon with Headers

COOJA Simulator

- A **network simulator** inside Contiki specifically designed for **Wireless Sensor Networks**.
- A highly useful tool for Contiki development as it allows developers to **test** their code and systems long before running it on the target hardware.

Get Started with Contiki

- Download Instant Contiki (2.2 GB)

 http://sourceforge.net/projects/contiki/files/Instant%20Contiki/

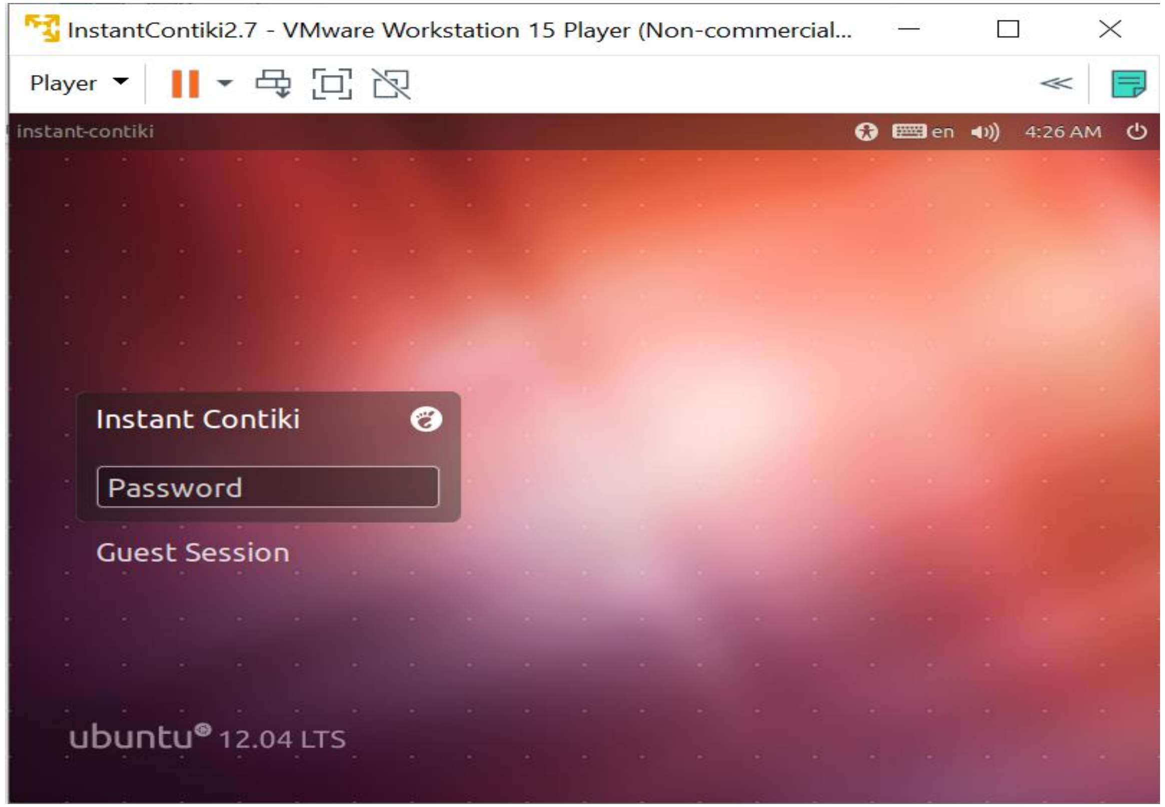
- Download VMWare Player (Windows) (It is free to download, but requires a registration)

http://www.vmware.com/go/downloadplayer/

  – For Mac Intel users, we use Virtual Box instead of VMWare
  – For Mac M1/M2 users, we use Vmware fusion (need to request license from UTS)
  – Last option: Install Windows on MAC ☹

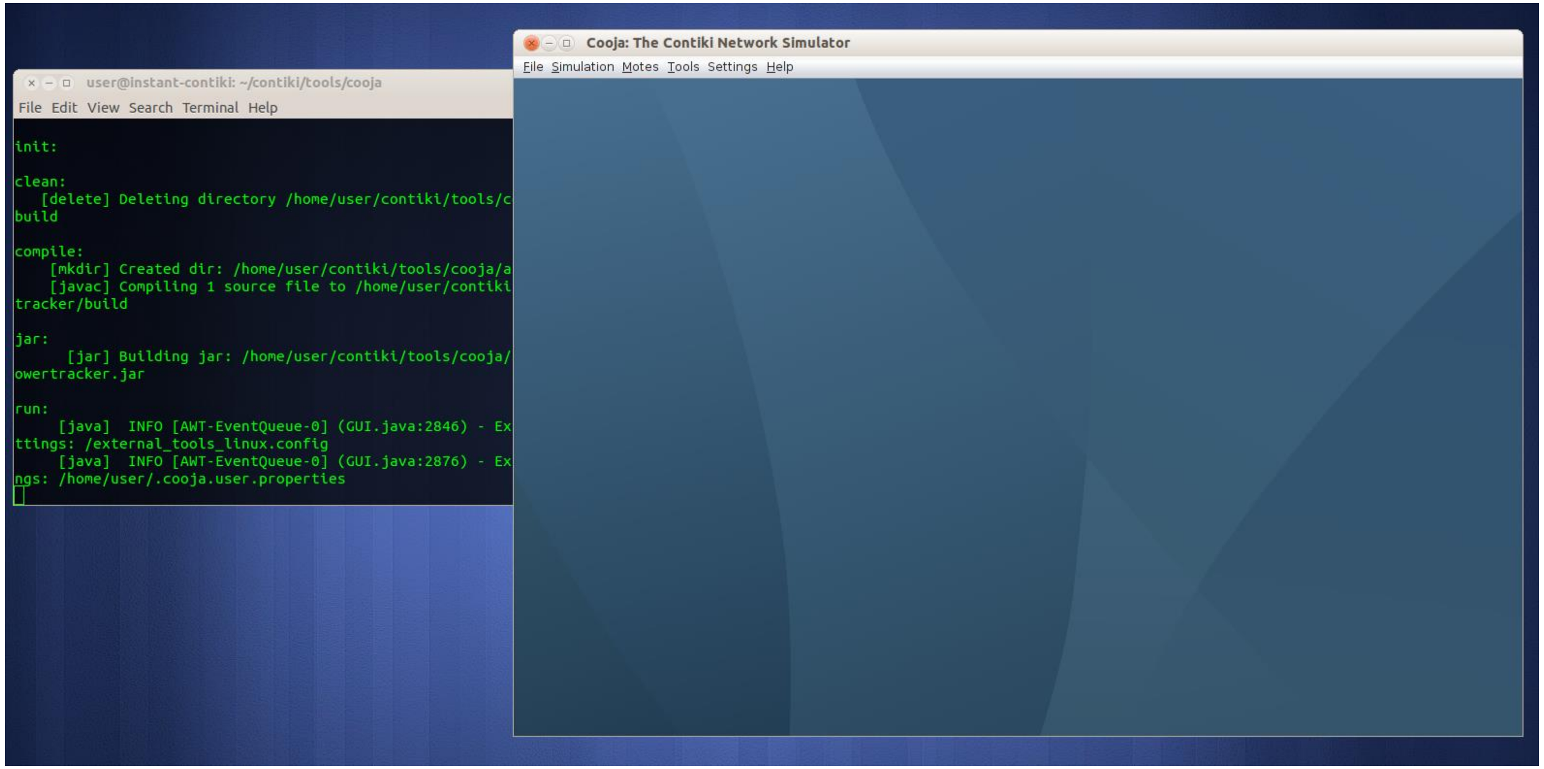
Get Started with Contiki

- Start Instant Contiki by running InstantContiki2.6.vmx. Wait for the virtual Ubuntu Linux boot up.
- Log into Instant Contiki. The password is **user**.



Start Cooja

- To start Cooja, first open a terminal window. In the terminal window, go to the Cooja directory: `cd contiki/tools/cooja`
- Start Cooja with the command: `ant run`

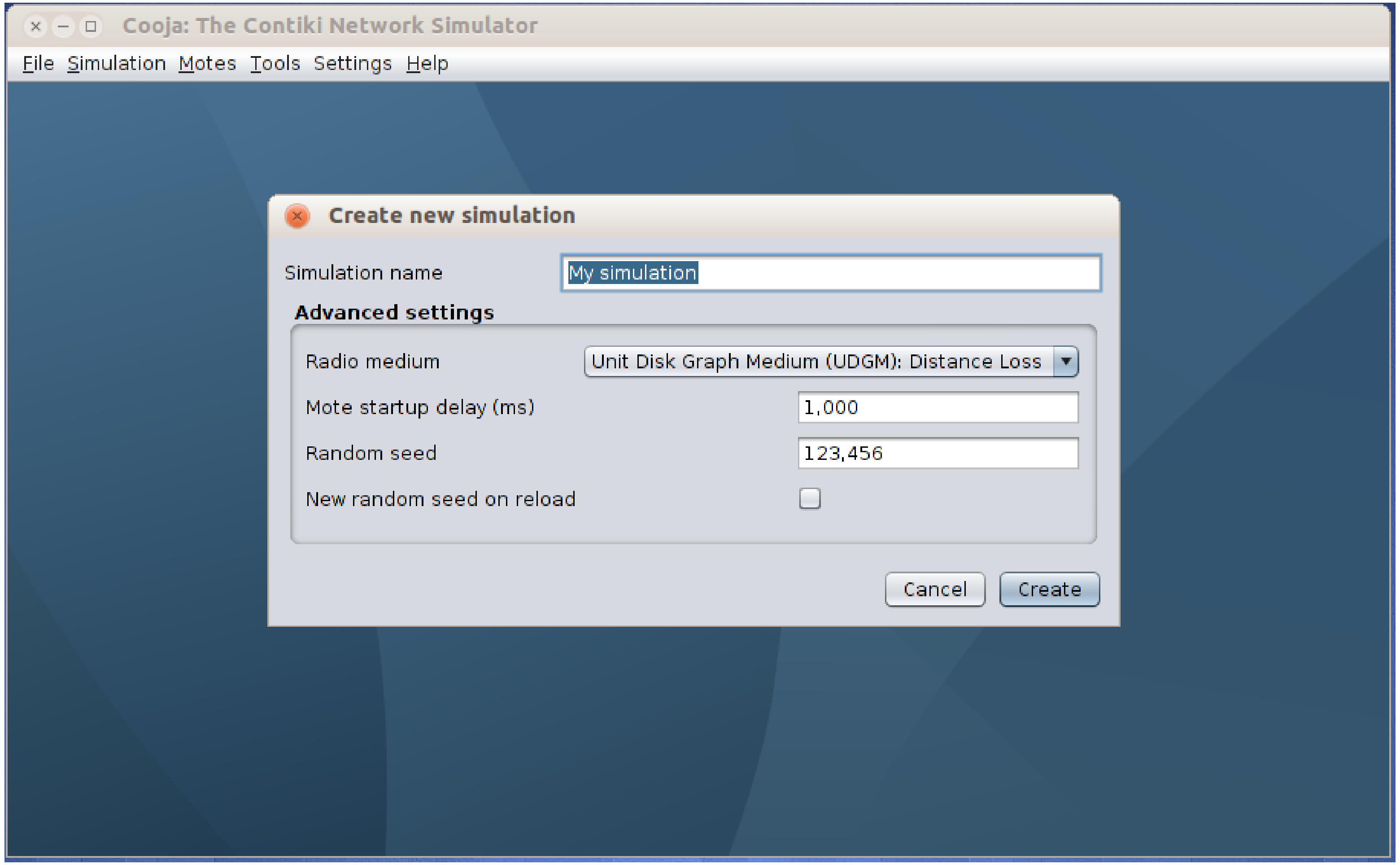


user@instant-contiki: ~/contiki/tools/cooja
File Edit View Search Terminal Help
Cooja: The Contiki Network Simulator
File Simulation Motes Tools Settings Help

```
init:

clean:
   [delete] Deleting directory /home/user/contiki/tools/c
build

compile:
    [mkdir] Created dir: /home/user/contiki/tools/cooja/a
    [javac] Compiling 1 source file to /home/user/contiki
tracker/build

jar:
      [jar] Building jar: /home/user/contiki/tools/cooja/
owertracker.jar

run:
     [java]  INFO [AWT-EventQueue-0] (GUI.java:2846) - Ex
ttings: /external_tools_linux.config
     [java]  INFO [AWT-EventQueue-0] (GUI.java:2876) - Ex
ngs: /home/user/.cooja.user.properties
```

Creating Simulation

- Click the File menu and click New simulation


Cooja: The Contiki Network Simulator
File Simulation Motes Tools Settings Help
Create new simulation
Cancel
Create

Simulation name: My simulation

Advanced settings

Radio medium: Unit Disk Graph Medium (UDGM): Distance Loss

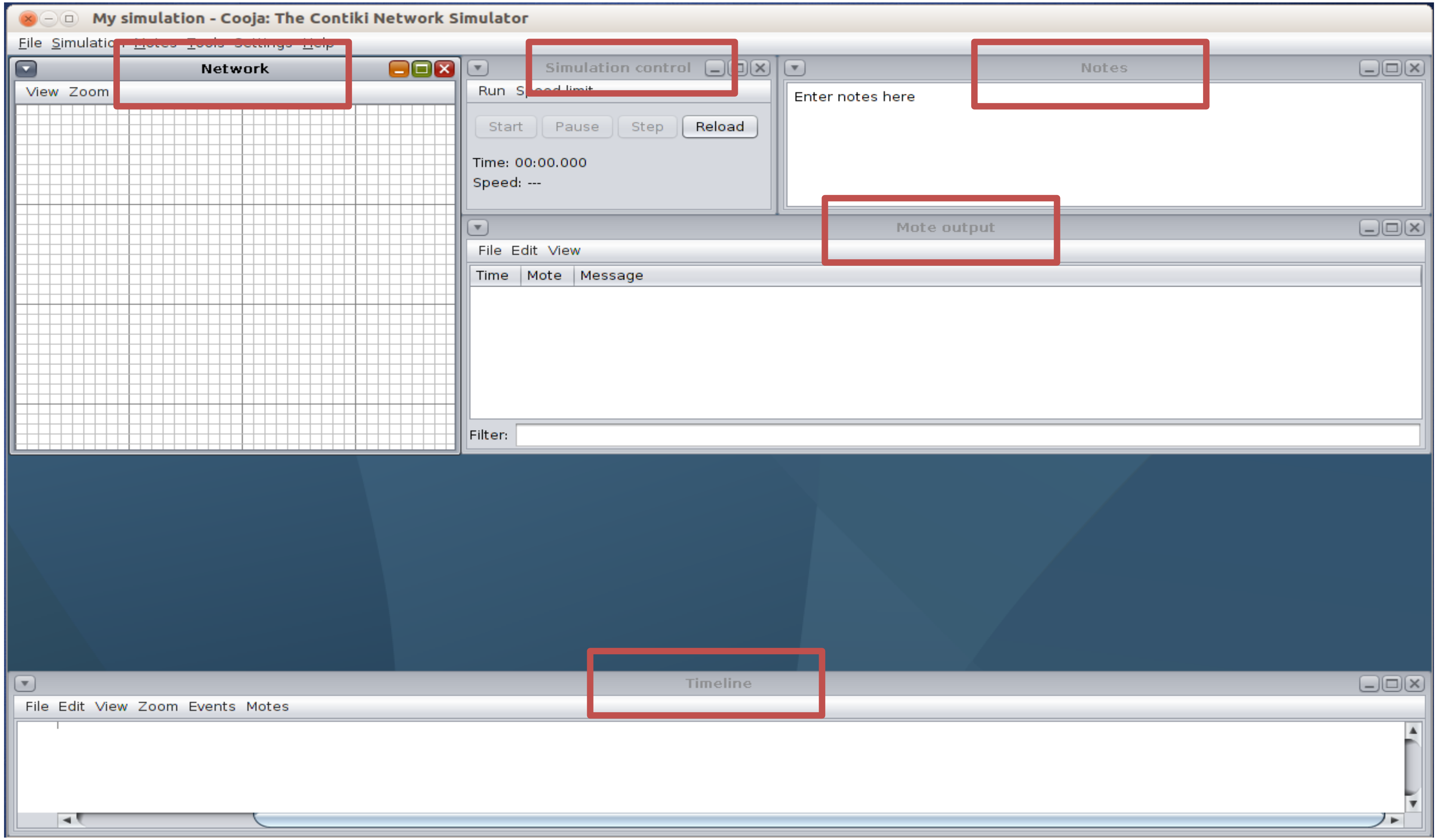
Mote startup delay (ms): 1,000

Random seed: 123,456

New random seed on reload: □

Creating Simulation

- Cooja brings up the new simulation

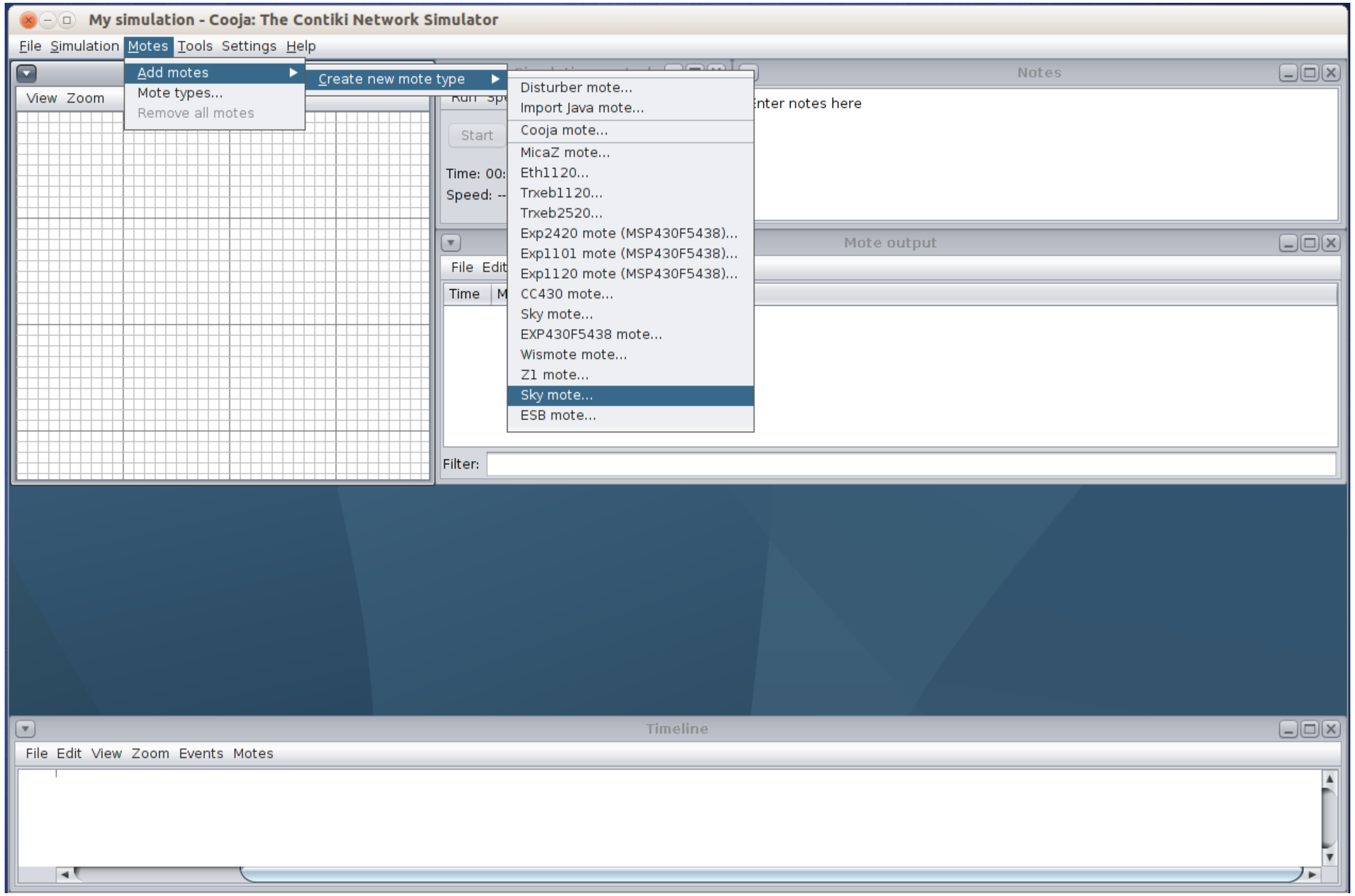


Creating Simulation

- The **Network window** (top left of the screen) shows all the motes in the simulated network
- The **Timeline window** (bottom of the screen) shows all communication events in the simulation over time
- The **Mote output window** (right side of the screen) shows all serial port printouts from all the motes.
- The **Notes window** on the top right is where we can put notes for our simulation.
- The **Simulation control** window is where we start, pause, and reload our simulation

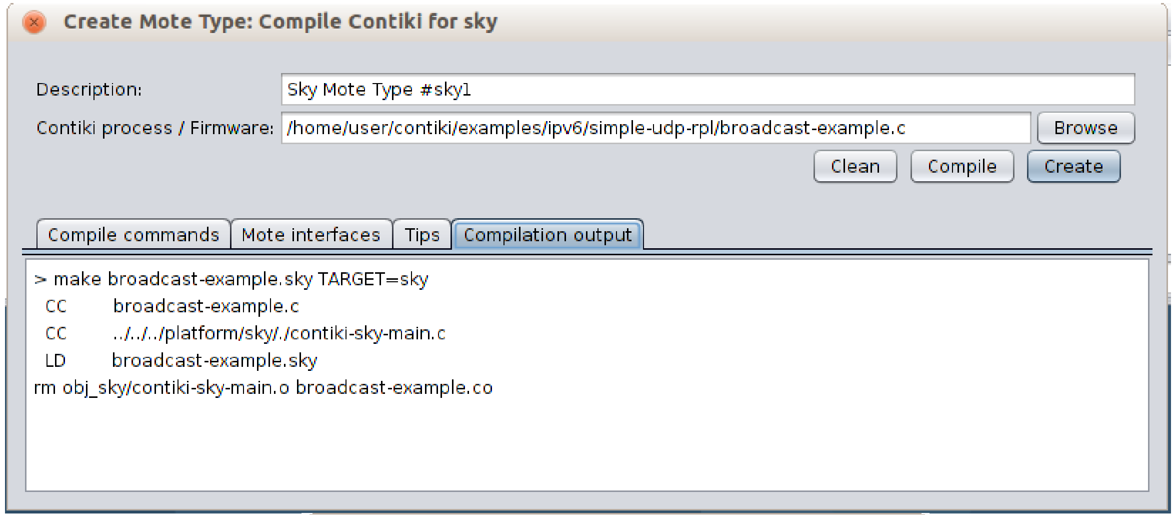
Creating Simulation

- Before we can simulate our network, we must add one or more motes



Creating Simulation

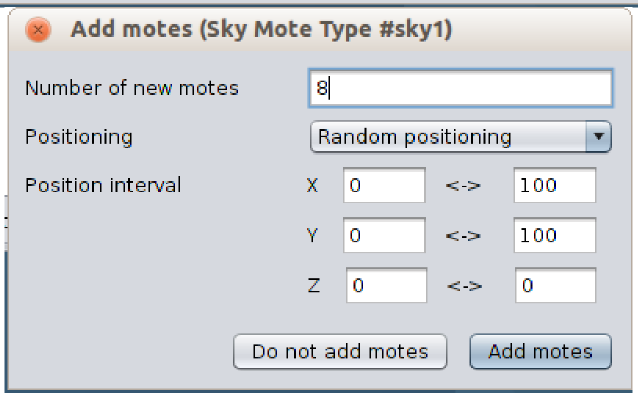
- Cooja opens the Create Mote Type dialog, in which we can choose a name for our mote type as well as the Contiki application that our mote type will run.



Description: Sky Mote Type #sky1

Contiki process / Firmware: /home/user/contiki/examples/ipv6/simple-udp-rpl/broadcast-example.c

```
> make broadcast-example.sky TARGET=sky
  CC        broadcast-example.c
  CC        ../../../platform/sky/./contiki-sky-main.c
  LD        broadcast-example.sky
rm obj_sky/contiki-sky-main.o broadcast-example.co
```

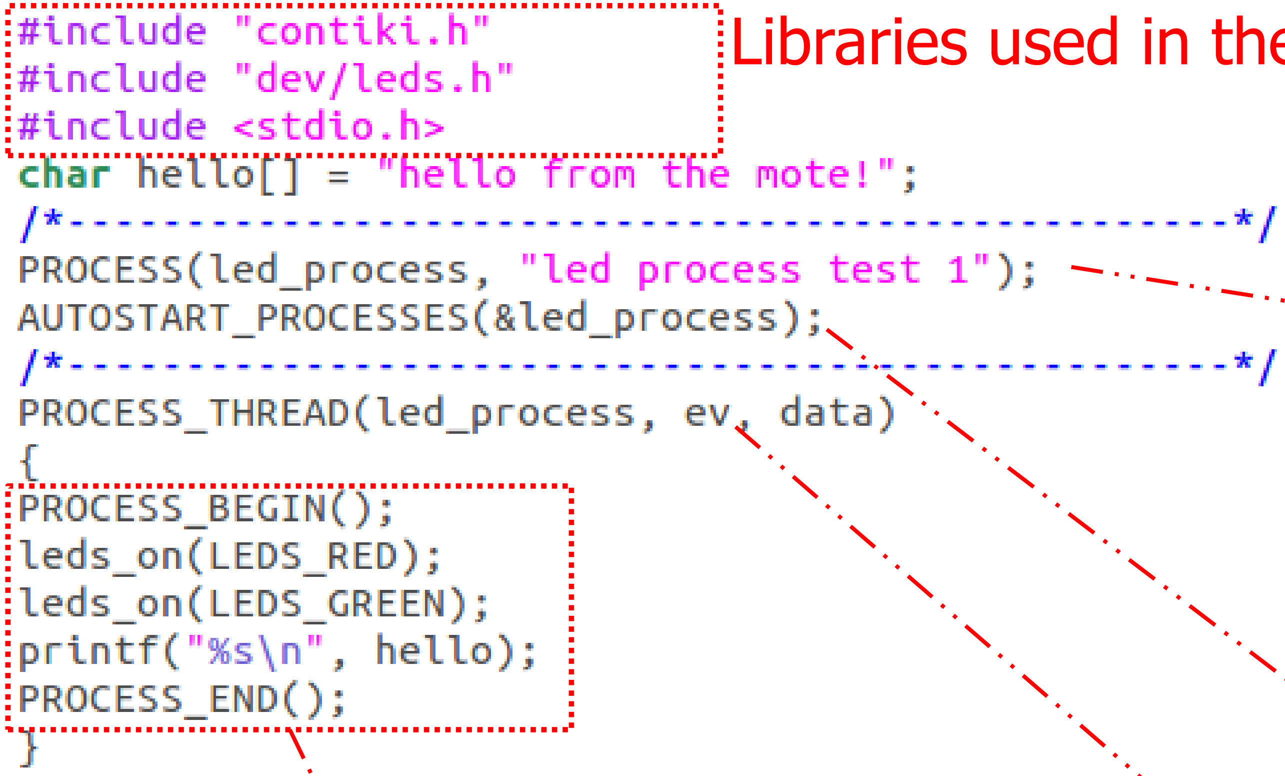


Number of new motes: 8

Positioning: Random positioning

Position interval:
X 0 <-> 100
Y 0 <-> 100
Z 0 <-> 0

Simple Example: Turn on LEDs on the Sky mote



```
#include "contiki.h"
#include "dev/leds.h"
#include <stdio.h>
char hello[] = "hello from the mote!";
/*---------------------------------------------------------------------------*/
PROCESS(led_process, "led process test 1");
AUTOSTART_PROCESSES(&led_process);
/*---------------------------------------------------------------------------*/
PROCESS_THREAD(led_process, ev, data)
{
PROCESS_BEGIN();
leds_on(LEDS_RED);
leds_on(LEDS_GREEN);
printf("%s\n", hello);
PROCESS_END();
}
```

Libraries used in the code

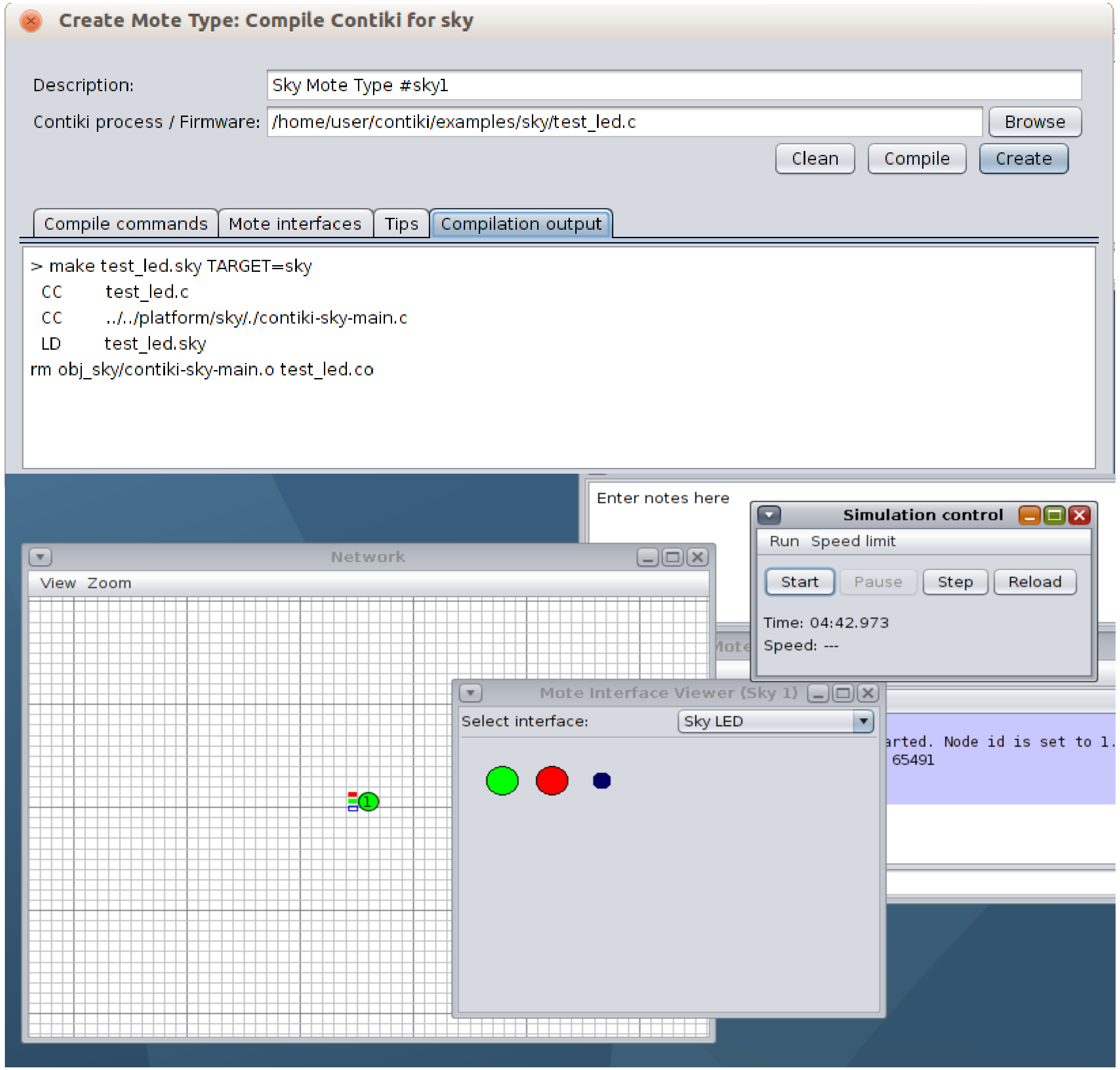
“led_process” is the name of the process and *“led process test 1”* is the readable name of the process when you print it to the terminal.

Tells Contiki to start that process when it finishes booting.

Inside the thread you begin the process, do what you want and finally end the process

You declare the content of the process in the process thread. You have the name of the process and callback functions (event handler and data handler).

Simple Example: Turn on LEDs on the Sky mote



Description: Sky Mote Type #sky1

Contiki process / Firmware: /home/user/contiki/examples/sky/test_led.c

```
> make test_led.sky TARGET=sky
  CC        test_led.c
  CC        ../../platform/sky/./contiki-sky-main.c
  LD        test_led.sky
rm obj_sky/contiki-sky-main.o test_led.co
```

Time: 04:42.973

Speed: ---

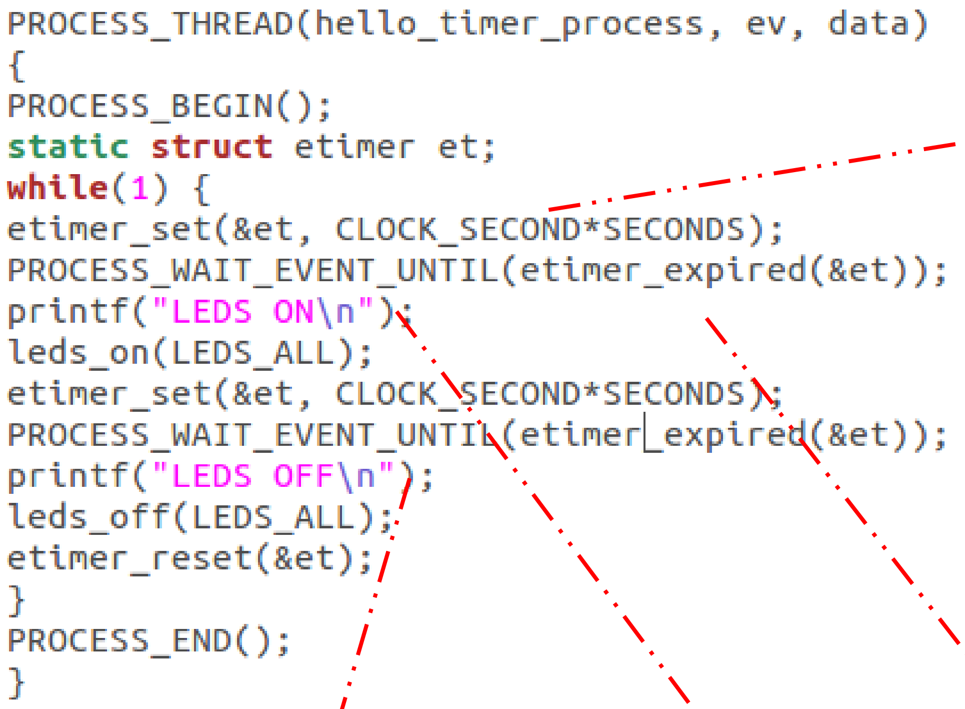
Select interface: Sky LED

```
arted. Node id is set to 1.
 65491
```

Blink Application - Timer

- Contiki OS provides 4 kinds of timers:
  - Simple timer: A simple ticker, the application should check manually if the timer has expired. More information at core/sys/timer.h .
  - Callback timer: When a timer expires, it can callback a given function. More information at core/sys/ctimer.h .
  - **Event timer**: Same as above, but instead of calling a function, when the timer expires, it posts an event signalling its expiration. More information at core/sys/etimer.h .
  - Real time timer: The real-time module handles the scheduling and execution of real-time tasks; there's only 1 timer available at the moment. More information at core/sys/rtimer.h

Blink Application - Timer



```
PROCESS_THREAD(hello_timer_process, ev, data)
{
PROCESS_BEGIN();
static struct etimer et;
while(1) {
etimer_set(&et, CLOCK_SECOND*SECONDS);
PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
printf("LEDS ON\n");
leds_on(LEDS_ALL);
etimer_set(&et, CLOCK_SECOND*SECONDS);
PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
printf("LEDS OFF\n");
leds_off(LEDS_ALL);
etimer_reset(&et);
}
PROCESS_END();
}
```

CLOCK_SECOND is a value related to the number of the microcontroller’s ticks per second. As Contiki runs on different platforms with different hardware, the value of CLOCK_SECOND also differs.

Turn on all leds

PROCESS_WAIT_EVENT_UNTIL() waits for the timer expired

Turn off all leds

Exercises

1. Switch on the LED when the button is pressed. Switch off the LED when the button is pressed again.
2. Blink the LED for a certain number of seconds.
3. A new application that starts only when the button is pressed and when the button is pressed again it stops.

References

1. Get Started with Contiki: http://www.contiki-os.org/start.html
2. Contiki Tutorial: http://anrg.usc.edu/contiki/index.php/Contiki_tutorials
3. A. Dunkels, B. Gronvall, and T. Voigt, “Contiki-a lightweight and flexible operating system for tiny networked sensors,” IEEE international conference on local computer networks, Tampa, FL, USA, Nov. 2004.

Thank
You
köszönöm
Kiitos
Vinaka
Maake
Asante
Shukria
Dhanyavadagalu
감사합니다
Dankscheen
спасибо
Kam Sah Hammida
شكرا
Manana
Dankon
Dank Je
Mauruuru
Biyan
Diolch i Chi
Terima Kasih
Taiku
Matondo
Blagodaram
Ngiyabonga
Dziekuje
Chokrane
Tack
Juspaxar
Arigato
Grazie
Mochchakkeram
Gracias
Khap
Paldies
Tingki
நன்றி
धन्यवाद
cảm ơn bạn
Bedankt
Ďakujem
Grazas
Di Ou Mèsi
Gratias Tibi
Ua Tsaug Rau Koj
Kia Ora
Kop Khun
Obrigado
Děkuji
Nirringrazzjak
ありがとう
Suksama
Nuwun
Matur
Rahmat
谢谢
хвала
Hvala
Welalin
Djiere Dieuf
Misaotra
Danke
Merci
Go Raibh Maith Agat
Tuke
Eskerrik Asko
Salamat
ขอขอบคุณคุณ
Najis