IT-SDN (v0.4.1): Installation Guide (for Linux 64 bits - Unbuntu 18.04.1 LTS - March, 2019)

Renan C. A. Alves[1], **Doriedson A. G. Oliveira**[1], **Gustavo N. Segura**[1], **Cintia B. Margi**[1*]

[1]Escola Politécnica – Universidade de São Paulo – São Paulo, Brazil

Contents

*C. B. Margi is supported by CNPq research fellowship #307304/2015-9.

1. Introduction

IT-SDN is a Software Defined Wireless Sensor Network (SDWSN) tool that is completely open and freely available, designed to be independent of the operating system and its functions. Although it is inspired by TinySDN [de Oliveira et al. 2014, de Oliveira and Margi 2016], we improved the architecture, protocols and implementation.

The IT-SDN framework version presented here comprises SDN-enabled WSN nodes developed under Contiki OS. We provide sample code for data generating nodes (named enabled-nodes throughout the document) and sink-nodes. The controller is developed in C and C++ with Qt (we call this software controller-pc). The controller-pc software must connect to a WSN node for communicating with the other nodes in the network. The node programmed with the interfacing software is called controller-node.

In order to compile and run IT-SDN nodes and controller, you need to follow these steps (described in the next sections):

2. Obtain Contiki OS;
3. Obtain IT-SDN;
4. Obtain the GCC toolchain for MSP430;
5. Obtain Qt framework;
6. IT-SDN compilation;
7. Simulating IT-SDN;
8. Running on hardware.

2. Requirements

2.1. Obtain Contiki OS

Download Contiki release 3.0 available at `https://github.com/contiki-os/contiki/releases/tag/3.0`

After downloading, place the contiki directory in the home folder `/home/USERNAME` and rename directory from contiki-3.0 to contiki.

Alternatively, you can get Contiki release 3.0 in the command line, as follows:

```
$ cd /home/USERNAME
$ wget https://github.com/contiki-os/contiki/archive/3.0.zip
$ unzip 3.0.zip
$ mv contiki-3.0 ~/contiki
```

You need to get MSPSim to emulate instruction level of MSP430 series micropocessor on Cooja (the Contiki Network Simulator). Download it from `https://github.com/contiki-os/mspsim` and unpack it in directory `/home/USERNAME/contiki/tools/mspsim/`.

To run Cooja, you will need Java JDK 8 as well as the ant build tool. If you need to install them, use the following commands in a terminal.

```
~$ sudo apt-get install ant
~$ sudo add-apt-repository ppa:webupd8team/java
~$ sudo apt-get update
~$ sudo apt-get install oracle-java8-set-default
```

To test if download and extraction succeeded, run the command `ant run` on folder `/home/USERNAME/contiki/tools/cooja`:

```
~/contiki/tools/cooja$ ant run
```

If everything was installed correctly, it will start with a blue empty window (Cooja: The Contiki Network Simulator) as shown in Figure 1.

You can get more information about Cooja at `https://github.com/contiki-os/contiki/wiki/An-Introduction-to-Cooja`.

2.2. Obtain IT-SDN

Download the IT-SDN framework from `http://www.larc.usp.br/~cbmargi/it-sdn`

When downloaded, extract and place the directory on `/home/USERNAME`.

2.3. Obtain the GCC toolchain for MSP430

We are using MSP430-based platforms (such as Tmote) and thus the following packages are needed: `binutils-msp430`, `gcc-msp430`, `msp430-libc`, `msp430mcu` and `mspdebug` to compile and run enabled-nodes and controller-node developed to ContikiOS.

You can run the following command to install all packages:

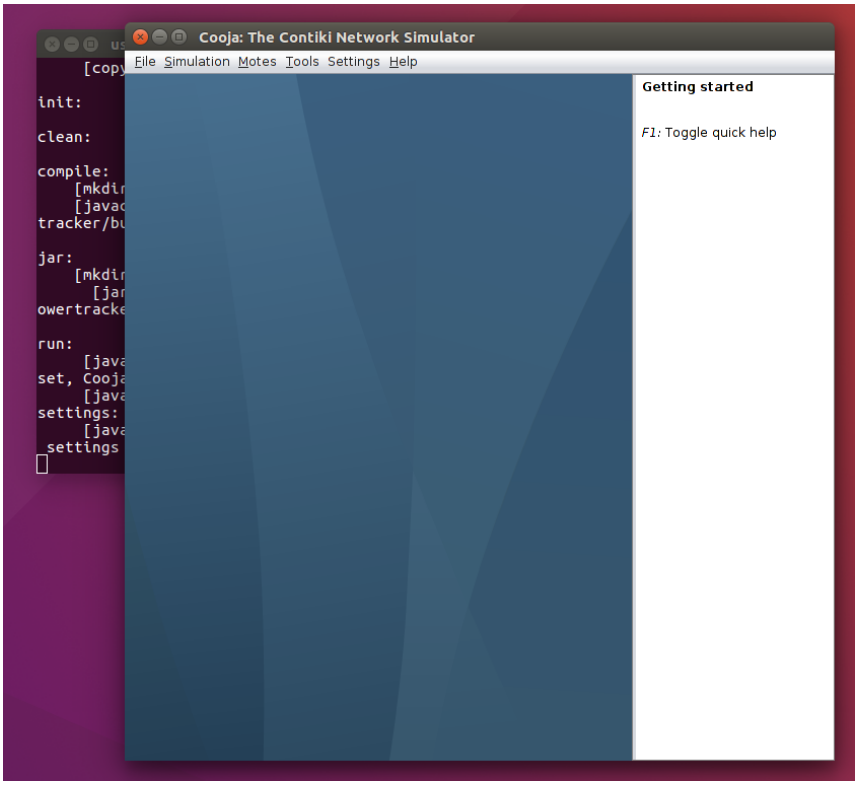

Figure 1. Cooja execution interface.

```
$ sudo apt-get install build-essential binutils-msp430 gcc-msp430 msp430-libc
msp430mcu mspdebug
```

You can get more information about MSP430-gcc at `https://github.com/jlhonora/mspgcc-install`.

If all previous steps succeed, you should be able to compile IT-SDN as instructed in Section 3.1.

2.4. Obtain Qt framework

To compile and run the `controller-pc` software you will need to download and install QT framework with Qt 5.9[1] available at `https://www.qt.io/download/`. Figure 2 shows Qt Creator interface.

You also need to install another package:

```
$ sudo apt-get install libgl1-mesa-dev
```

[1] the latest patch version as of the writing of this document is 5.9.7

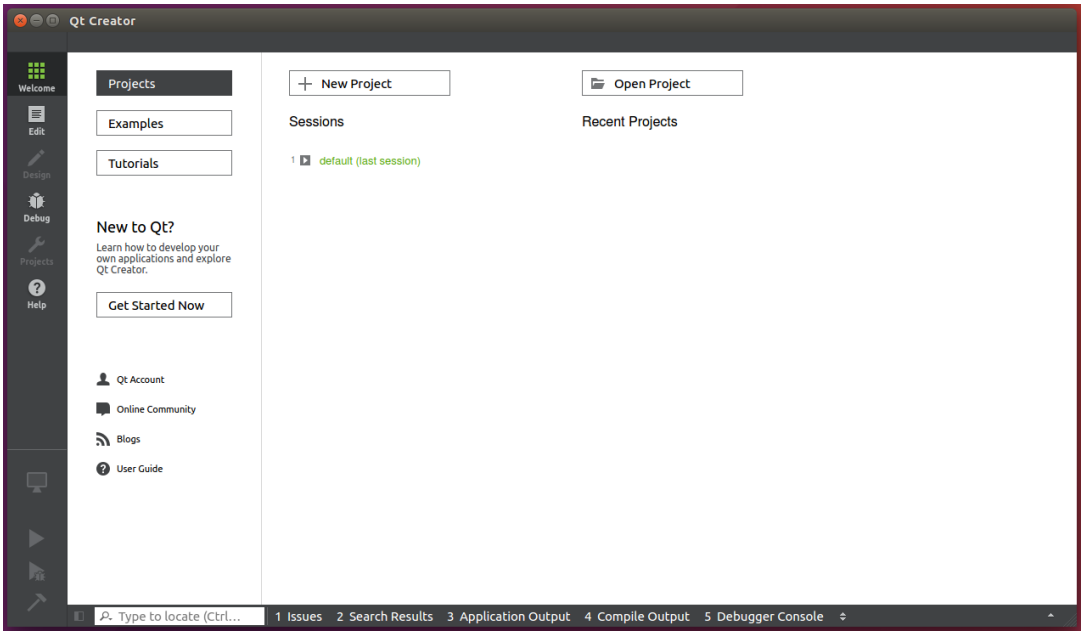

Figure 2. Qt Creator opened.

3. IT-SDN compilation

In the next sections, we describe how to compile and execute IT-SDN.

3.1. Compile enabled-nodes and controller-node

To compile enabled-node and the controller-node source code example you should change the first line `Contiki=...` on the `Makefile_enabled_node` and `Makefile_controller_node`, in folder `it-sdn/applications`, to indicates the Contiki source files as shown below:

```
CONTIKI=/home/USERNAME/contiki
```

And to compile go to the applications directory and use the compilation script provided as follows. Verify if script has the appropriate permission to execute.

```
$ cd ~/it-sdn/applications/
$ ./compile.sh
```

You can change the target platform typing the platform as script parameter of the `compile.sh` file. Example:

```
$ ./compile.sh sky
```

The target is defined to sky (telosb) if the script is called without parameter.

In order to show that the compilation succeeded, we used `size *.sky` as the last command of the compile.sh script. If compilation succeeded, you will see the size of the three firmwares (enabled-node, sink-node and controller-node), as shown in Figure 3.

3.2. Compile controller-pc

Before compiling the controller-pc, you need to indicate Contiki path. To do so, create a file named `controller-pc.pro.contiki` and save on folder

```
  CC        /home/user/contiki/core/net/mac/framer.c
  CC        /home/user/contiki/core/net/queuebuf.c
  CC        /home/user/contiki/core/net/nbr-table.c
  CC        /home/user/contiki/core/net/packetbuf.c
  CC        /home/user/contiki/core/net/netstack.c
  CC        /home/user/contiki/core/net/linkaddr.c
  CC        /home/user/contiki/core/net/mac/contikimac/contikimac.c
  CC        /home/user/contiki/core/net/mac/contikimac/contikimac-framer.c
  CC        /home/user/contiki/core/net/mac/cxmac/cxmac.c
  CC        /home/user/contiki/core/net/llsec/nullsec.c
  CC        /home/user/contiki/core/net/llsec/anti-replay.c
  CC        /home/user/contiki/core/net/llsec/ccm-star-packetbuf.c
  CC        /home/user/contiki/core/net/llsec/noncoresec/noncoresec.c
  CC        symbols.c
  AR        contiki-sky.a
  CC        controller-node.c
  CC        /home/user/contiki/platform/sky/./contiki-sky-main.c
  LD        controller-node.sky
rm controller-node.co obj_sky/contiki-sky-main.o
   text    data     bss     dec     hex filename
  32048     162    8704   40914    9fd2 controller-node.sky
  45038     196    8768   54002    d2f2 enabled-node.sky
  44626     196    8746   53568    d140 sink-node.sky
user@it-sdn:~/it-sdn/applications$
```

Figure 3. Compiling enabled-node, sink-node and controller-node.

`/home/USERNAME/it-sdn/controller-server/controller-pc` with the following content:

```
CONTIKI=/home/USERNAME/contiki
```

To compile and run controller-pc, open the Qt Creator and click on button [Open Project]. Navigate to the path `/home/USERNAME/it-sdn/controller-server/controller-pc` and select the project `controller-pc.pro`.

You will see the Configure Project window (depicted in Figure 4), click on [Configure Project].

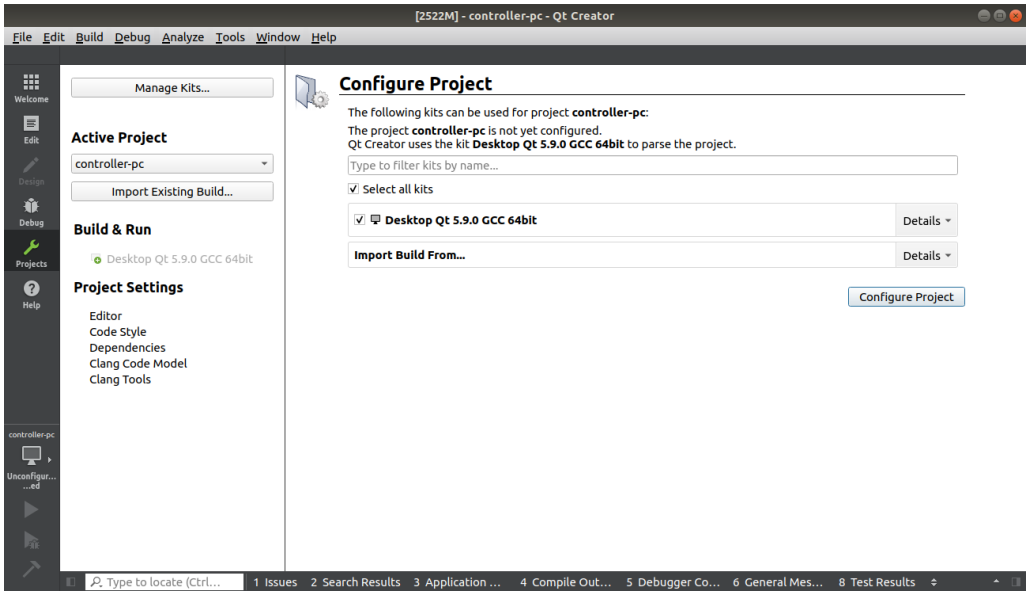

Figure 4. Configuring project kit to controller-pc.

Now you are ready to compile the controller-pc. Click on menu Build→Build All.

4. Simulating IT-SDN

4.1. Running controller-pc

After compiling, click on menu Build→Run on Qt Creator to run controller-pc. You will see the controller-pc window as shown in Figure 5.

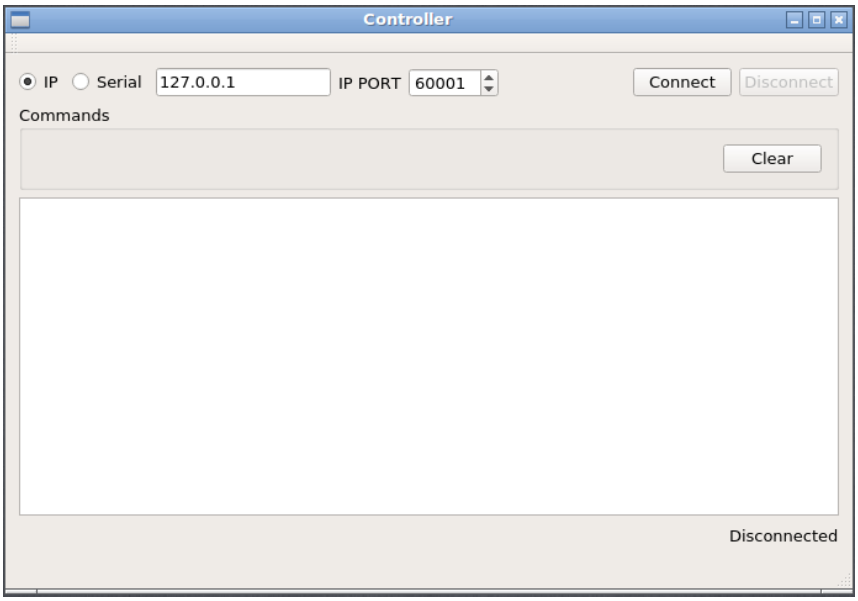

Figure 5. First window for controller-pc.

4.2. Starting Cooja

In the terminal window, go to the Cooja directory and then start Cooja as follows.

```
$ cd ~/contiki/tools/cooja/
$ ant run
```

After Cooja opened, go to the File menu and click Open simulation → Browse... Navigate to `/home/USERNAME/it-sdn/simulation/` and open file `ContikiSDNSimulation.csc`, as shown in Figure 6.

4.3. Connecting controller-pc to controller-node

Once Cooja simulation loaded, you need to connect the controller-pc to controller-node. On controller-pc window (Figure 5), click [Connect] button. Once they are connected, the controller interface will change as depicted in Figure 7.

4.4. Running Cooja Simulation

After controller-pc has connected on Cooja simulation, click on [Start] to initialize the simulation, as shown in Figure 8. The Mote output window contains serial output from all nodes. IT-SDN nodes print messages to ease the computation of network metrics and statistics, as well as messages regarding node status and message transmission. These features can be activated by defining, respectively, the macros SDN_METRIC or DEBUG_SDN in the application makefile.

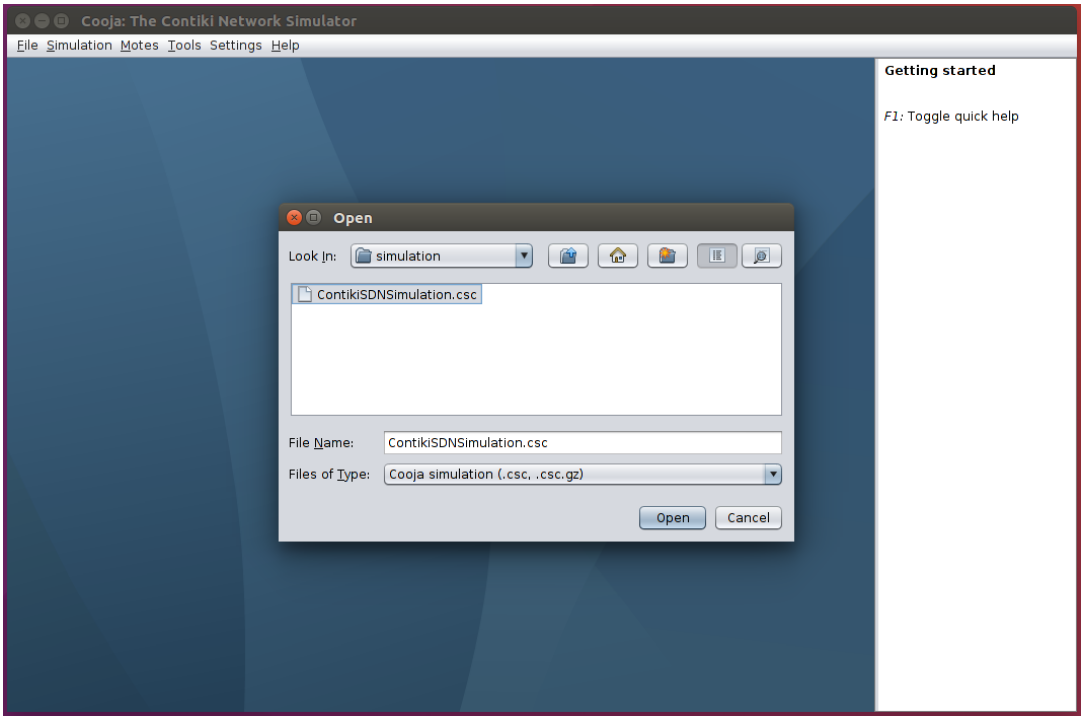

Figure 6. Opening Cooja simulation.

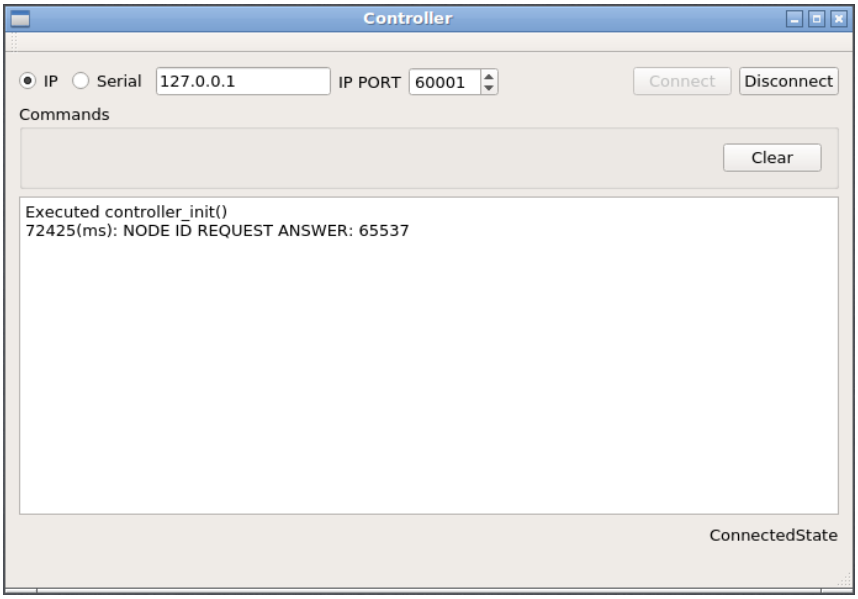

Figure 7. Connecting the controller-pc to controller-node.

5. Running on hardware

In this section we describe how you can execute IT-SDN in a testbed, i.e., run an application using IT-SDN framework on actual devices. For the description, we assume you will be using TelosB motes.

Notice: We have tested the framework both in the TelosB and SensorTag[2] devices.

[2] `http://www.ti.com/tool/TIDC-CC2650STK-SENSORTAG`

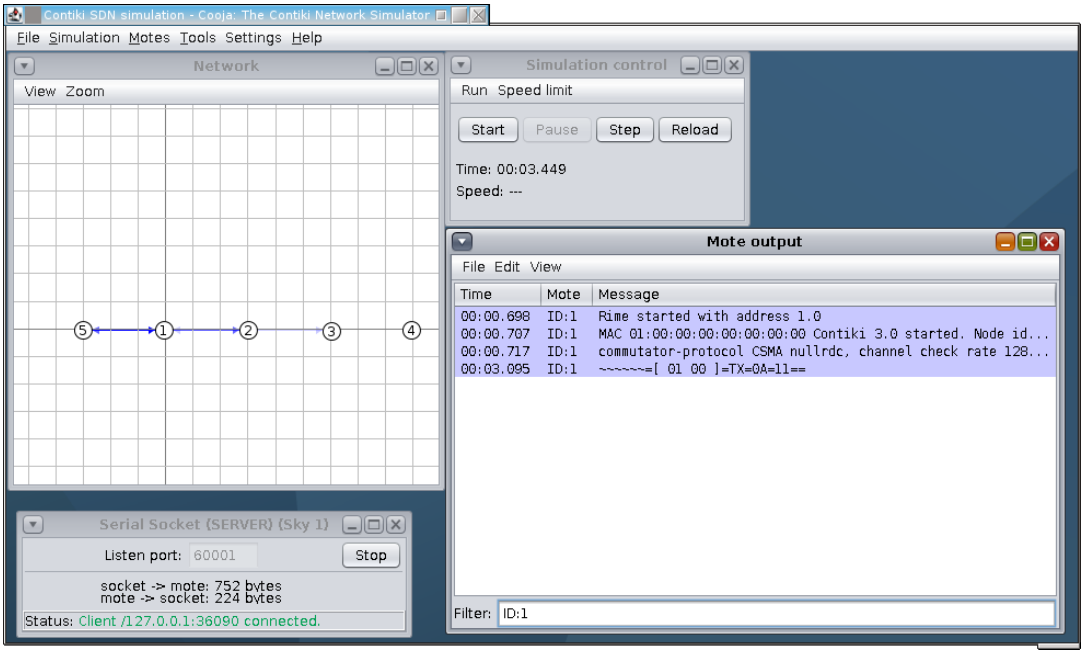

Figure 8. Starting Cooja simulation.

5.1. Uploading enabled-node and sink-node firmwares to TelosB motes

To upload the IT-SDN enabled-node and sink-node firmwares to a TelosB, you must connect the device to a USB port of your PC.
After connecting the TelosB on USB port you can check the TelosB port address through of the command:

```
$ motelist
```

The Figure 9 depict the output command. You can observe the address on `Device` column.

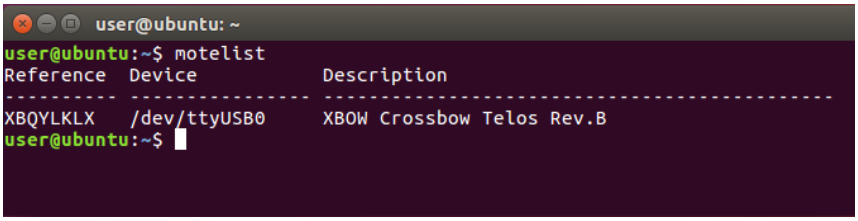

Figure 9. Example to terminal output from motelist command.

You can install the `motelist` tool by typing `sudo apt-get install tinyos-tools`.
To upload firmware through the serial port, you have to add your user to group `dialout` with the following command:

```
$ sudo adduser USERNAME dialout
```

Here, we consider that the mote TelosB is connected on address `/dev/ttyUSB0`, as shown in Figure 9.
You can upload enabled-node firmware with the following command:

```
$ make -f Makefile_enabled_node TARGET=sky clean MOTES=/dev/ttyUSB0
enabled-node.upload
```

In case showing permission denied for serial port, make sure you are a member of `dialout` group in the current session. You may have to start a new shell session or logout/login.

The `MOTES=/dev/ttyUSB0` parameter is only necessary if you have more than one connected mote on USB to indicates which mote should receives the firmware by upload. To upload sink-node firmware type the following command:

```
$ make -f Makefile_enabled_node TARGET=sky clean MOTES=/dev/ttyUSB0
sink_node.upload
```

Repeat the process for as many TelosB devices as you want to be SDN enabled-nodes and sink-nodes in the testbed.

5.2. Uploading enabled-node and sink-node firmwares to SensorTag CC2650STK

To upload the IT-SDN enabled-node and sink-node firmware to a SensorTag, you must download and install a flash programmer for Texas Instruments devices. In this case we chose UniFlash v4, which can be downloaded from `http://processors.wiki.ti.com/index.php/Category:CCS_UniFlash`.

Then, you must compile the firmware using Contiki 3.x. To obtain this Contiki version, you can clone it using the command shown below:

```
$ git clone --recursive https://github.com/contiki-os/contiki.git
```

We use the recursive command to be sure we also clone the libraries required by the SensorTag.

Before compiling, you must change the Contiki variable path in `/home/USERNAME/it-sdn/applications/Makefile_enabled_node` to the Contiki 3.x path. Then, you can compile the firmware using the commands:

```
$ cd ~/it-sdn/applications/
$ make clean -f Makefile_enabled_node TARGET=srf06-cc26xx
BOARD=sensortag/cc2650 enabled-node.bin CPU_FAMILY=cc26xx
```

For the sink the command you must use is:

```
$ cd ~/it-sdn/applications/
$ make clean -f Makefile_enabled_node TARGET=srf06-cc26xx
BOARD=sensortag/cc2650 sink-node.bin CPU_FAMILY=cc26xx
```

The next step is to open UniFlash 4.5 and then connect the Sensortag to a serial port. If UniFlash is configured in automatic mode, it will detect the SensorTag, such as shown in Figure 10. Otherwise, you will have to choose the platform model manually. Then, you can press button *Start*, choose the `enabled-node.bin` or `sink-node.bin` file, and load the image.

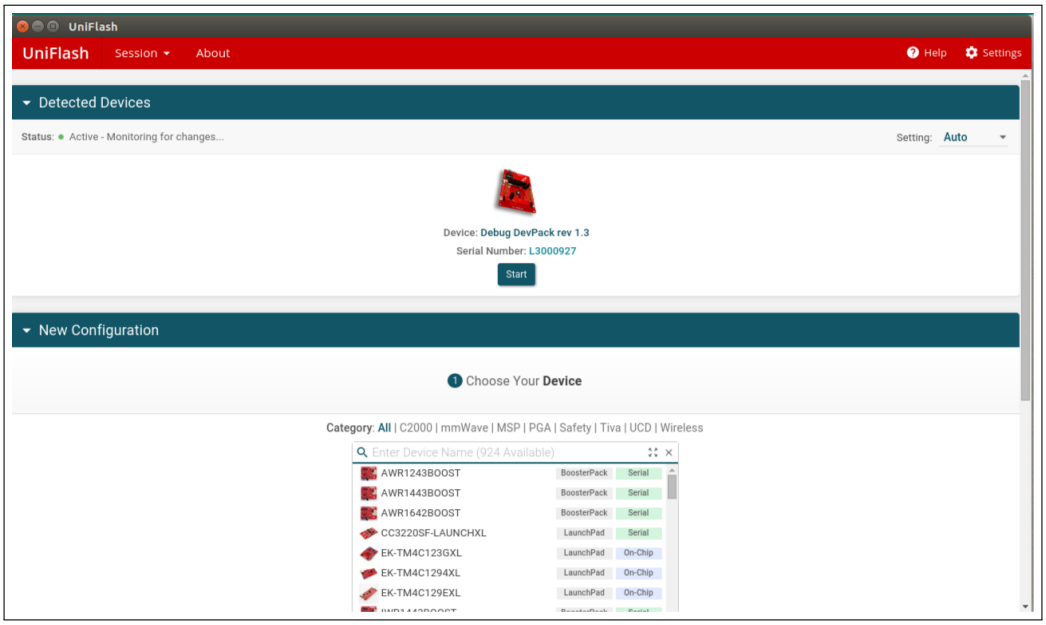

Figure 10. UniFlash v4.5.

5.3. Running controller-node on TelosB mote

Open a terminal window and go to the applications directory:

```
$ cd ~/it-sdn/applications/
```

To upload to the mote, type the following command:

```
$ make -f Makefile_controller_node TARGET=sky clean MOTES=/dev/ttyUSB2
controller-node.upload
```

Lastly, run the controller-pc and connect to mote controller-node connected to a USB port of your PC, as shown in Figure 11.

5.4. Running controller-node on SensorTag CC2650STK

You must change the Contiki variable path in `Makefile_controller_node` to the Contiki 3.x path. Then, you can compile the firmware using the commands:

```
$ cd ~/it-sdn/applications/
$ make clean -f Makefile_controller_node TARGET=srf06-cc26xx
BOARD=sensortag/cc2650 controller-node.bin CPU_FAMILY=cc26xx
```

The next step is to open upload the firmware to the node using UniFlash, such as explained in Section 5.2 for enabled-node and sink-node firmwares.

Lastly, run the controller-pc and connect the SensorTag to a USB port. Select the serial option in the controller-pc, set it to `/dev/ttyACM0` and press `Connect`.

5.5. Using TelosB and SensorTag CC2650STK in the same network

In the case of working with a heterogeneous network using TelosB mote and SensorTag, you must go to `/home/USERNAME/it-sdn/applications/project-conf.h` and add the next line:

```
#define RF_CORE_CONF_CHANNEL 26
```

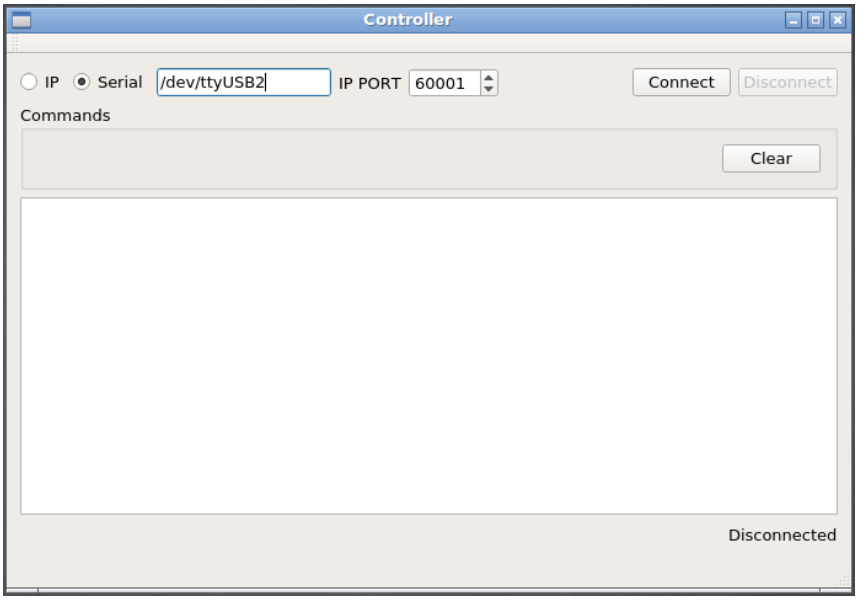

Figure 11. Controller-pc configured to connect on telosb.

References

de Oliveira, B. T. and Margi, C. B. (2016). TinySDN: Enabling TinyOS to Software-Defined Wireless Sensor Networks. In *XXXIV Simpósio Brasileiro de Redes de Computadores*, pages 1229–1237.

de Oliveira, B. T., Margi, C. B., and Gabriel, L. B. (2014). TinySDN: Enabling multiple controllers for software-defined wireless sensor networks. In *Communications (LATINCOM), 2014 IEEE Latin-America Conference on*, pages 1–6.