Advanced Network Security Lab
Session 2: IEEE 802.15.4 MAC-Layer Security

Dominik Roy George
`d.r.george@tue.nl`

Savio Sciancalepore
`s.sciancalepore@tue.nl`

February 28, 2024

1 Introduction

This second lab session provides you with insights into the *MAC-Layer Security* of the reference OS used for lab sessions, namely, *contiki-ng*.
The main objective of this second session is to let you visualize the configurations of the *IEEE 802.15.4 MAC-Layer Security*, how to capture live traffic in a Wireless Sensor Network (WSN), and observe the impact of various MAC-layer security configurations on the network behavior.
For this session, we will be working with three example programs of the *contiki-ng*. We need one simple IoT node, one sniffer node and a border router. The one node will be flashed with the *hello-world* program of the *contiki-ng* and the second node will be flashed with the *sensniff* program, while the border router will build on the example program *rpl-border-router*. This document provides you with the required instructions. We kindly request the students follow the instructions during the lab session.

1.1 Scenario

For the second lab session, you will be working in a **group of three members**. For each group, we will provide three *Sensortags CC2650*. In Fig. 1, we illustrate the scenario you will deploy during the lab session.

One SensorTag will behave as a *simple node* and will be communicating with the border-router. The second SensorTag will act as a *sniffer*. The third SensorTag, connected to a Laptop, will behave as the border-router of the network. At the same time, the simple node and the sniffer can be connected to the laptops of the other group members.

In Tab. 1.1, we provide *group ids* and associated *PAN IDs* and *Channels*. We provide these ids to prevent mutual interferences during the lab sessions. Therefore, we kindly request each group not to configure the devices to non-associated ids.

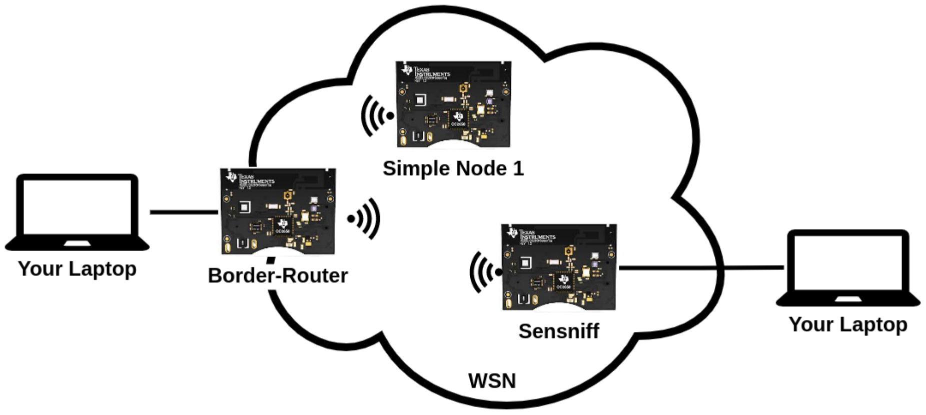


Figure 1: Scenario for Lab Session 2.

To fully enjoy this lab session, we provide instructions in four sections: (i) Security Configuration, (ii) Wireshark Configuration, (iii) Sensniff, and (iv) Deployment.

2 IEEE 802.15.4 Security Configuration

You need to apply the following steps for all the *Sensortags CC2650* which act as *simple node* and as *border-router* in your network.

2.1 Step 1: Configuration

Open the file *project-conf.h* with any editor of your choice. Next, copy the following code inside:

Listing 1: project-conf.h

```
#ifndef PROJECT_CONF_H_
#define PROJECT_CONF_H_
//COPY FROM HERE AND ADD THE CODE in the existing project-conf.h
//LOGGING
#define LOG_CONF_LEVEL_RPL LOG_LEVEL_DBG
#define LOG_CONF_LEVEL_TCPIP LOG_LEVEL_WARN
#define LOG_CONF_LEVEL_IPV6 LOG_LEVEL_WARN
#define LOG_CONF_LEVEL_6LOWPAN LOG_LEVEL_DBG
#define LOG_CONF_LEVEL_MAC LOG_LEVEL_DBG
#define LOG_CONF_LEVEL_FRAMER LOG_LEVEL_INFO
#define TSCH_LOG_CONF_PER_SLOT 0

#undef IEEE802154_CONF_DEFAULT_CHANNEL
```

Group ID	PAN ID	CHANNEL
1	0xABCD	11
2	0xABCE	12
3	0xABCF	13
4	0xABD0	14
5	0xABD1	15
6	0xABD2	16
7	0xABD3	17
8	0xABD4	18
9	0xABD5	19
10	0xABD6	20
11	0xABD7	21
12	0xABD8	22
13	0xABD9	23
14	0xABDA	24
15	0xABDB	25
16	0xABDC	11
17	0xABDD	12
18	0xABDE	13
19	0xABDF	14
20	0xABE0	15
21	0xABE1	16
22	0xABE2	17
23	0xABE3	18

```
#define IEEE802154_CONF_DEFAULT_CHANNEL YOUR CHANNEL
#undef IEEE802154_CONF_PANID
#define IEEE802154_CONF_PANID YOUR PANID

//SECURITY
#define LLSEC802154_CONF_ENABLED 1
#define CSMA_CONF_LLSEC_SECURITY_LEVEL 5
#define CSMA_CONF_LLSEC_KEY_ID_MODE 2
//COPY END
#endif
```

☞ We remind you to change the variable *IEEE802154_CONF_DEFAULT_CHANNEL* and *IEEE802154_CONF_PANID* according to your *group id* in Tab. 1.1.

2.2 Steps 2 and 3

After updating the *project-conf.h*, you need to execute the following command to compile the program, generating an executable file for the *Sensortag CC2650*:

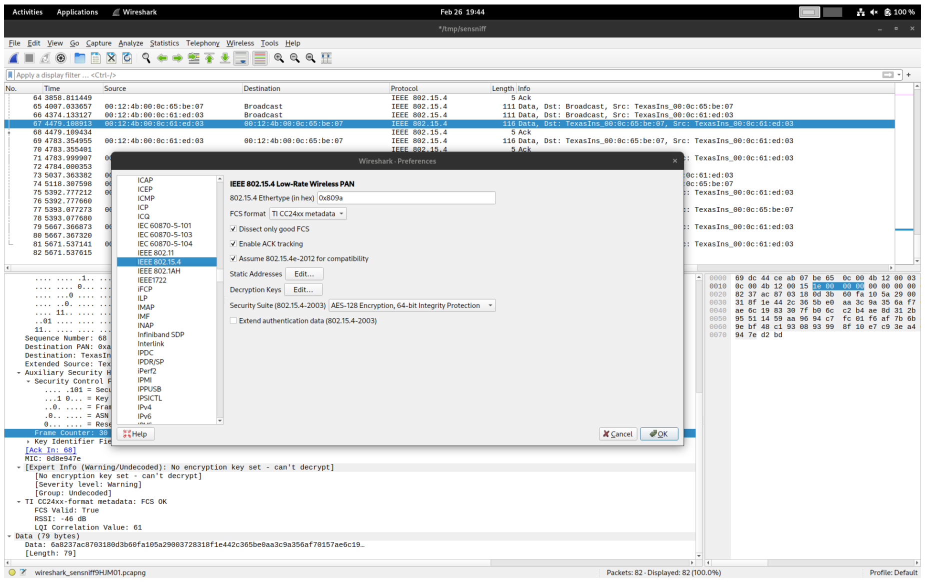
Listing 2: Compile

```
$ sudo make TARGET=cc26x0-cc13x0 BOARD=sensortag/cc2650 clean
$ sudo make TARGET=cc26x0-cc13x0 BOARD=sensortag/cc2650
```

Steps 2 and 3 are equivalent, as presented in the first lab session. For flashing the executable with UNIFLASH, select the executables in the base folder of the individual project folders, such as *hello-world.cc26x0-cc13x0*, *border-router.cc26x0-cc13x0*.

3 Wireshark Configuration

In this lab session, we use Wireshark as a tool for capturing live traffic of WSN. First, you need to define preferences in Wireshark. Therefore, open Wireshark's preferences and select *'TI CC24xx FCS format'* under Protocols of *IEEE 802.15.4*. To allow you to verify correct configuration, we provide a Wireshark screenshot, shown in Fig. 2.



IEEE 802.15.4 Low-Rate Wireless PAN
802.15.4 Ethertype (in hex) 0x809a
FCS format TI CC24xx metadata
[✓] Dissect only good FCS
[✓] Enable ACK tracking
[✓] Assume 802.15.4e-2012 for compatibility
Static Addresses Edit...
Decryption Keys Edit...
Security Suite (802.15.4-2003) AES-128 Encryption, 64-bit Integrity Protection
[ ] Extend authentication data (802.15.4-2003)

```
No. Time        Source                   Destination              Protocol      Length Info
 64 3858.811449                                                   IEEE 802.15.4    5 Ack
 65 4007.033657 00:12:4b:00:0c:65:be:07  Broadcast                IEEE 802.15.4  111 Data, Dst: Broadcast, Src: TexasIns_00:0c:65:be:07
 66 4374.133127 00:12:4b:00:0c:61:ed:03  Broadcast                IEEE 802.15.4  111 Data, Dst: Broadcast, Src: TexasIns_00:0c:61:ed:03
 67 4479.108913 00:12:4b:00:0c:61:ed:03  00:12:4b:00:0c:65:be:07  IEEE 802.15.4  116 Data, Dst: TexasIns_00:0c:65:be:07, Src: TexasIns_00:0c:61:ed:03
 68 4479.109434                                                   IEEE 802.15.4    5 Ack
 69 4783.354955 00:12:4b:00:0c:61:ed:03  00:12:4b:00:0c:65:be:07  IEEE 802.15.4  116 Data, Dst: TexasIns_00:0c:65:be:07, Src: TexasIns_00:0c:61:ed:03
 70 4783.355401                                                   IEEE 802.15.4    5 Ack
 71 4783.999907 00                                                 : TexasIns_00:0c:61:ed:03
 72 4784.000353
 73 5037.363382 00                                                 0c:61:ed:03
 74 5118.307598 00                                                 0c:65:be:07
 75 5392.777212 00                                                 : TexasIns_00:0c:61:ed:03
 76 5392.777660
 77 5393.077273 00                                                 : TexasIns_00:0c:65:be:07
 78 5393.077680
 79 5667.366873 00                                                 : TexasIns_00:0c:61:ed:03
 80 5667.367320
 81 5671.537141 00                                                 : TexasIns_00:0c:61:ed:03
 82 5671.537615
```

```
    .... .... .1.. ...
    .... .... 0... ...
    .... ...0 .... ...
    .... ..0. .... ...
    .... 11.. .... ...
    ..01 .... .... ...
    11.. .... .... ...
  Sequence Number: 68
  Destination PAN: 0xa
  Destination: TexasIn
  Extended Source: Tex
▾ Auxiliary Security H
  ▾ Security Control F
      .... .101 = Secu
      ...1 0... = Key
      ..0. .... = Fran
      .0.. .... = ASN
      0... .... = Rese
    Frame Counter: 30
  ▸ Key Identifier Fie
  [Ack In: 68]
  MIC: 0d8e947e
▾ [Expert Info (Warning/Undecoded): No encryption key set - can't decrypt]
    [No encryption key set - can't decrypt]
    [Severity level: Warning]
    [Group: Undecoded]
▾ TI CC24xx-format metadata: FCS OK
    FCS Valid: True
    RSSI: -46 dB
    LQI Correlation Value: 61
▾ Data (79 bytes)
  Data: 6a8237ac8703180d3b60fa105a29003728318f1e442c365be0aa3c9a356af70157ae6c19…
  [Length: 79]
```

```
0000  69 dc 44 ce ab 07 be 65  0c 00 4b 12 00 03
0010  0c 00 4b 12 00 15 1e 00  00 00 00 00 00 00
0020  82 37 ac 87 03 18 0d 3b  60 fa 10 5a 29 00
0030  31 8f 1e 44 2c 36 5b e0  aa 3c 9a 35 6a f7
0040  ae 6c 19 83 30 7f b0 6c  c2 b4 ae 8d 31 2b
0050  95 51 14 59 aa 96 94 c7  fc 01 f6 af 7b 6b
0060  9e bf 48 c1 93 08 93 99  8f 10 e7 c9 3e a4
0070  94 7e d2 bd
```

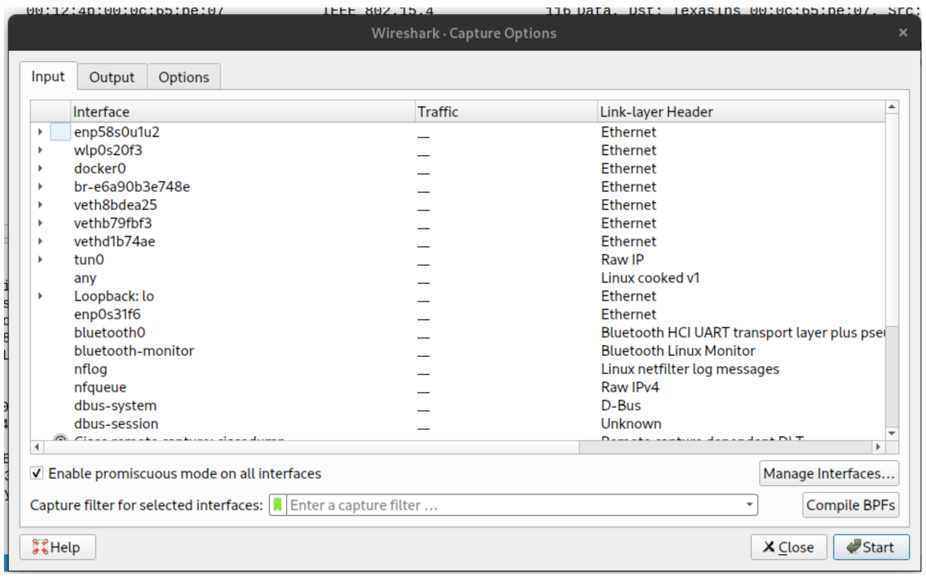
wireshark_sensniff9HJM01.pcapng   Packets: 82 · Displayed: 82 (100.0%)   Profile: Default

Figure 2: Wireshark Configuration.

Next, we define in Wireshark a *pipe*, which appears as an interface to capture the live traffic. Go to the tab *Capture* and select *options*, then a window will pop up, as shown in Fig. 3.

Further, click on the button *Manage Interfaces...* in the window. Afterwards, a window opens, where you can select the *Pipe* tab to add a new pipe as an interface, as shown in Fig. 4.

Click now on the plus button to create a new pipe, type */tmp/sensniff* and save it. For more details, we refer to [1].



Wireshark · Capture Options

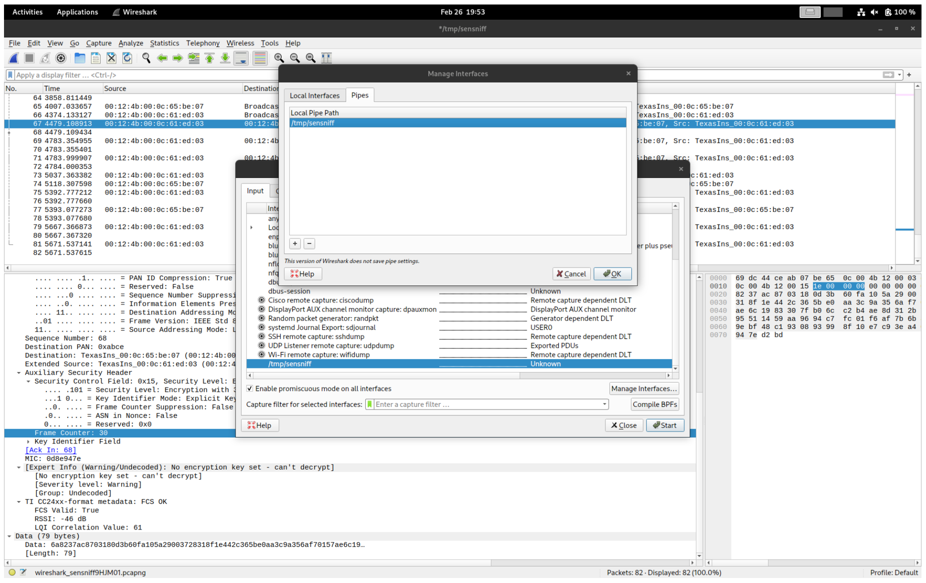
Input | Output | Options

Interface	Traffic	Link-layer Header
enp58s0u1u2	—	Ethernet
wlp0s20f3	—	Ethernet
docker0	—	Ethernet
br-e6a90b3e748e	—	Ethernet
veth8bdea25	—	Ethernet
vethb79fbf3	—	Ethernet
vethd1b74ae	—	Ethernet
tun0	—	Raw IP
any	—	Linux cooked v1
Loopback: lo	—	Ethernet
enp0s31f6	—	Ethernet
bluetooth0	—	Bluetooth HCI UART transport layer plus pse
bluetooth-monitor	—	Bluetooth Linux Monitor
nflog	—	Linux netfilter log messages
nfqueue	—	Raw IPv4
dbus-system	—	D-Bus
dbus-session	—	Unknown

✓ Enable promiscuous mode on all interfaces

Capture filter for selected interfaces: Enter a capture filter …

Figure 3: Wireshark Capture Options.



Manage Interfaces

Local Interfaces | Pipes

Local Pipe Path
/tmp/sensniff

This version of Wireshark does not save pipe settings.

Interface	Traffic	Link-layer Header
dbus-session	—	Unknown
Cisco remote capture: ciscodump	—	Remote capture dependent DLT
DisplayPort AUX channel monitor capture: dpauxmon	—	DisplayPort AUX channel monitor
Random packet generator: randpkt	—	Generator dependent DLT
systemd Journal Export: sdjournal	—	USER0
SSH remote capture: sshdump	—	Remote capture dependent DLT
UDP Listener remote capture: udpdump	—	Exported PDUs
Wi-Fi remote capture: wifidump	—	Remote capture dependent DLT
/tmp/sensniff	—	Unknown

✓ Enable promiscuous mode on all interfaces

Capture filter for selected interfaces: Enter a capture filter …

Figure 4: Wireshark Manage Interfaces.

4 Sensniff

Sensniff is a software application, which acts as a sniffer in the WSN to capture the traffic flowing through a given interface. Navigate to the *sensniff* folder:

Listing 3: sensniff.

```
$ cd /your-working-folder/contiki-ng/examples/sensniff
```

To compile and flash the program on the SensorTag, follow steps 2 and 3 of the *simple node configuration* from the first lab session. Flash the executable from the sensniff folder, i.e., *sensniff.cc26x0-cc13x0* on a Sensortag device.

5 Network Deployment

In the previous sections, you prepared the nodes by flashing the executables for the associated roles described in Fig. 1. Here, we assume all the devices are correctly flashed.

First, connect the border-router to the host system and navigate to *tools/serial-io* to execute the following command:

Listing 4: Execute tunslip6.

```
$ sudo ./tunslip6 -s /dev/ttyACM0 fd00::1/64
```

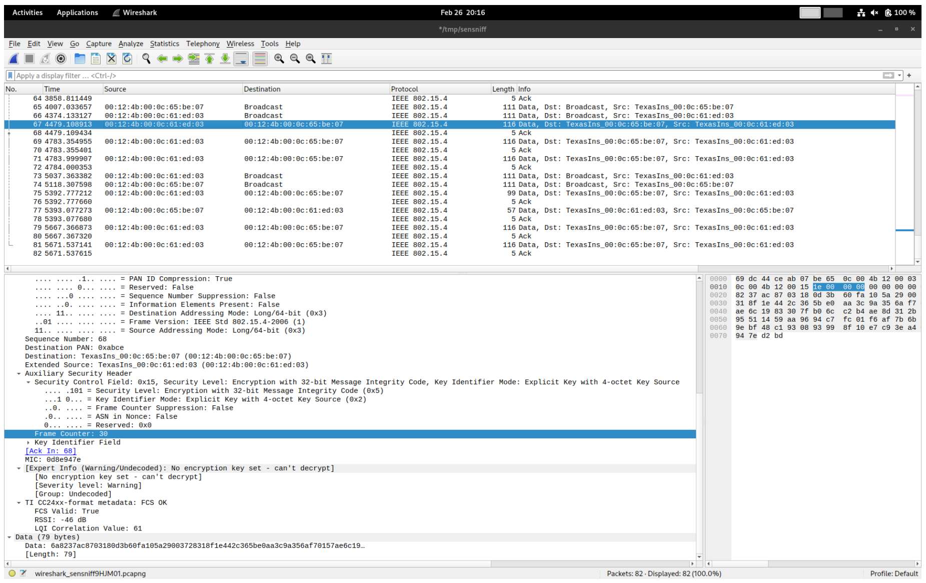
Next, connect/start the *sniffer* by executing the following command in *tools/sensniff* folder, contained into the *contiki-ng* folder:

Listing 5: Sensniff.

```
$ sudo python3 sensniff.py -b 460800 -d /dev/ttyACM0
```

After starting the sniffer, type YOUR CHANNEL number in the interactive terminal of sensniff. Otherwise, the sensniff does not know on which channel it should sniff. Next, connect the *simple node*. Finally, open Wireshark, navigate to *Capture* and options, and select the previously configured interface. It will take some time for the IEEE 802.15.4 packets to appear in Wireshark (wait approx. 2 minutes). If every step was successfully executed, you should start seeing the live traffic from the network, as shown in Fig. 5.

Take some time to familiarize with the content of the logged packets, trying to recognize the protocols and headers discussed in the frontal lectures.


Activities
Applications
Wireshark
Feb 26 20:16
100 %
*/tmp/sensniff
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
No.   Time          Source                    Destination               Protocol       Length Info
   64 3858.811449                                                       IEEE 802.15.4       5 Ack
   65 4007.033657   00:12:4b:00:0c:65:be:07   Broadcast                 IEEE 802.15.4     111 Data, Dst: Broadcast, Src: TexasIns_00:0c:65:be:07
   66 4374.133127   00:12:4b:00:0c:61:ed:03   Broadcast                 IEEE 802.15.4     111 Data, Dst: Broadcast, Src: TexasIns_00:0c:61:ed:03
   67 4479.108913   00:12:4b:00:0c:61:ed:03   00:12:4b:00:0c:65:be:07   IEEE 802.15.4     116 Data, Dst: TexasIns_00:0c:65:be:07, Src: TexasIns_00:0c:61:ed:03
   68 4479.109434                                                       IEEE 802.15.4       5 Ack
   69 4783.354955   00:12:4b:00:0c:61:ed:03   00:12:4b:00:0c:65:be:07   IEEE 802.15.4     116 Data, Dst: TexasIns_00:0c:65:be:07, Src: TexasIns_00:0c:61:ed:03
   70 4783.355401                                                       IEEE 802.15.4       5 Ack
   71 4783.999907   00:12:4b:00:0c:61:ed:03   00:12:4b:00:0c:65:be:07   IEEE 802.15.4     116 Data, Dst: TexasIns_00:0c:65:be:07, Src: TexasIns_00:0c:61:ed:03
   72 4784.000353                                                       IEEE 802.15.4       5 Ack
   73 5037.363382   00:12:4b:00:0c:61:ed:03   Broadcast                 IEEE 802.15.4     111 Data, Dst: Broadcast, Src: TexasIns_00:0c:61:ed:03
   74 5118.307598   00:12:4b:00:0c:65:be:07   Broadcast                 IEEE 802.15.4     111 Data, Dst: Broadcast, Src: TexasIns_00:0c:65:be:07
   75 5392.777212   00:12:4b:00:0c:61:ed:03   00:12:4b:00:0c:65:be:07   IEEE 802.15.4      99 Data, Dst: TexasIns_00:0c:65:be:07, Src: TexasIns_00:0c:61:ed:03
   76 5392.777660                                                       IEEE 802.15.4       5 Ack
   77 5393.077273   00:12:4b:00:0c:65:be:07   00:12:4b:00:0c:61:ed:03   IEEE 802.15.4      57 Data, Dst: TexasIns_00:0c:61:ed:03, Src: TexasIns_00:0c:65:be:07
   78 5393.077680                                                       IEEE 802.15.4       5 Ack
   79 5667.366873   00:12:4b:00:0c:61:ed:03   00:12:4b:00:0c:65:be:07   IEEE 802.15.4     116 Data, Dst: TexasIns_00:0c:65:be:07, Src: TexasIns_00:0c:61:ed:03
   80 5667.367320                                                       IEEE 802.15.4       5 Ack
   81 5671.537141   00:12:4b:00:0c:61:ed:03   00:12:4b:00:0c:65:be:07   IEEE 802.15.4     116 Data, Dst: TexasIns_00:0c:65:be:07, Src: TexasIns_00:0c:61:ed:03
   82 5671.537615                                                       IEEE 802.15.4       5 Ack
```

```
      .... .... .1.. .... = PAN ID Compression: True
      .... .... 0... .... = Reserved: False
      .... ...0 .... .... = Sequence Number Suppression: False
      .... ..0. .... .... = Information Elements Present: False
      .... 11.. .... .... = Destination Addressing Mode: Long/64-bit (0x3)
      ..01 .... .... .... = Frame Version: IEEE Std 802.15.4-2006 (1)
      11.. .... .... .... = Source Addressing Mode: Long/64-bit (0x3)
    Sequence Number: 68
    Destination PAN: 0xabce
    Destination: TexasIns_00:0c:65:be:07 (00:12:4b:00:0c:65:be:07)
    Extended Source: TexasIns_00:0c:61:ed:03 (00:12:4b:00:0c:61:ed:03)
  ▾ Auxiliary Security Header
    ▾ Security Control Field: 0x15, Security Level: Encryption with 32-bit Message Integrity Code, Key Identifier Mode: Explicit Key with 4-octet Key Source
        .... .101 = Security Level: Encryption with 32-bit Message Integrity Code (0x5)
        ...1 0... = Key Identifier Mode: Explicit Key with 4-octet Key Source (0x2)
        ..0. .... = Frame Counter Suppression: False
        .0.. .... = ASN in Nonce: False
        0... .... = Reserved: 0x0
      Frame Counter: 30
    ▸ Key Identifier Field
    [Ack In: 68]
    MIC: 0d8e947e
  ▾ [Expert Info (Warning/Undecoded): No encryption key set - can't decrypt]
      [No encryption key set - can't decrypt]
      [Severity level: Warning]
      [Group: Undecoded]
  ▾ TI CC24xx-format metadata: FCS OK
      FCS Valid: True
      RSSI: -46 dB
      LQI Correlation Value: 61
▾ Data (79 bytes)
    Data: 6a8237ac8703180d3b60fa105a29003728318f1e442c365be0aa3c9a356af70157ae6c19…
    [Length: 79]
```

```
0000  69 dc 44 ce ab 07 be 65  0c 00 4b 12 00 03
0010  0c 00 4b 12 00 15 1e 00  00 00 00 00 00 00
0020  82 37 ac 87 03 18 0d 3b  60 fa 10 5a 29 00
0030  31 8f 1e 44 2c 36 5b e0  aa 3c 9a 35 6a f7
0040  ae 6c 19 83 30 7f b0 6c  c2 b4 ae 8d 31 2b
0050  95 51 14 59 aa 96 94 c7  fc 01 f6 af 7b 6b
0060  9e bf 48 c1 93 08 93 99  8f 10 e7 c9 3e a4
0070  94 7e d2 bd
```

```
wireshark_sensniff9HJM01.pcapng        Packets: 82 · Displayed: 82 (100.0%)        Profile: Default
```

Figure 5: Wireshark IEEE 802.15.4 Traffic.

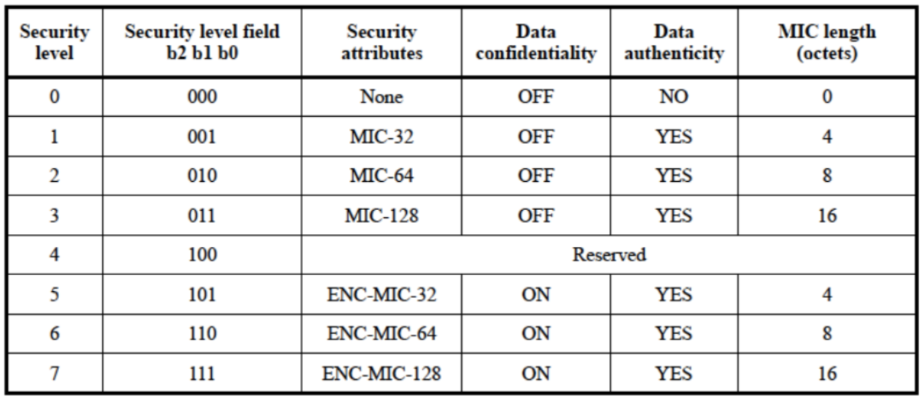
6 Playing with IEEE 802.15.4 Security Parameters

During the lecture on IEEE 802.15.4 Security, we introduced different *security levels* and *key identifier modes* associated with IEEE 802.15.4 packets. We hereby provide two figures where you can recall the configurations regarding the security level (Tab.6) and key mode (Tab.7).

You can now change the *securityLevel* and *keyIdMode* of IEEE 802.15.4 packets delivered by the devices involved in your local IoT network. Then, you can capture the packets via Wireshark and understand the differences between various security configurations.

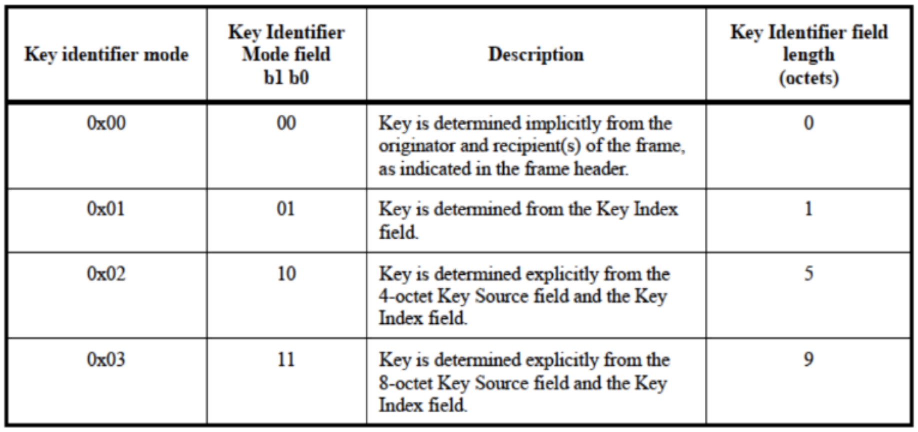
For changing the security level and the key identifier mode, see the code provided in Sec. 2, and especially the one related to the security part. We encourage you to visualize the effect of the change of the *securityLevel* and/or *keyMode* on the size of the packets exchanged in your network. For instance, you can take note of the size of a packet when *securityLevel=1*, *securityLevel=2*, and *securityLevel=3*, and match them with Tab. 6.
Then, you can do the same by setting *keyMode=1*, *keyMode=2* and *keyMode=3*.

You can also add your own key for both devices by adding in *project-conf.h* of both projects (*rpl-border-router and hello-world*) the following code:



Security level	Security level field b2 b1 b0	Security attributes	Data confidentiality	Data authenticity	MIC length (octets)
0	000	None	OFF	NO	0
1	001	MIC-32	OFF	YES	4
2	010	MIC-64	OFF	YES	8
3	011	MIC-128	OFF	YES	16
4	100	Reserved			
5	101	ENC-MIC-32	ON	YES	4
6	110	ENC-MIC-64	ON	YES	8
7	111	ENC-MIC-128	ON	YES	16

Figure 6: Security Level.



Key identifier mode	Key Identifier Mode field b1 b0	Description	Key Identifier field length (octets)
0x00	00	Key is determined implicitly from the originator and recipient(s) of the frame, as indicated in the frame header.	0
0x01	01	Key is determined from the Key Index field.	1
0x02	10	Key is determined explicitly from the 4-octet Key Source field and the Key Index field.	5
0x03	11	Key is determined explicitly from the 8-octet Key Source field and the Key Index field.	9

Figure 7: Key Mode.

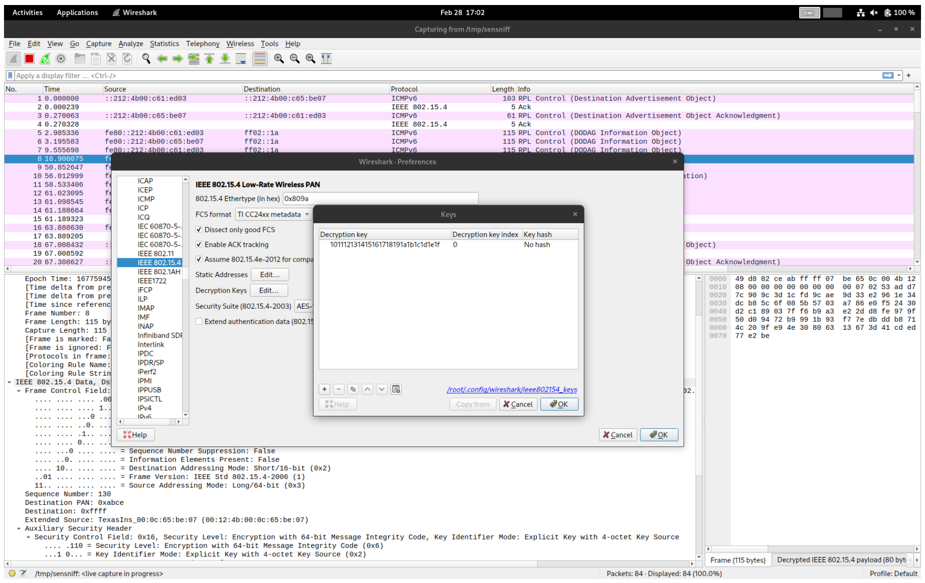
Listing 6: Execute tunslip6.

```
#define CSMA_CONF_LLSEC_DEFAULT_KEY0 {0x11,0x22,0x33,0x44,0x55,0
    x66,0x77,0x88,0x99,0xaa,0xbb,0xcc,0xdd,0xee,0xff,0x00}
```

In addition, you have to add your own defined key in Wireshark, to allow the cited sniffing tool to decrypt the frames. Therefore, open Wireshark's preferences and click on *'Decryption Keys'*, under Protocols of *IEEE 802.15.4*, as shown in Fig. 8. Moreover, click on the plus button to add your key, as shown in Fig. 9, with the default *key index:0* and *No hash*. Finally, restart all the devices by unplugging and plugging them all again. Afterwards, you should be able to see decrypted packets.

As another interesting attempt, you might change a byte in the key that you load into Wireshark, and verify that you are not able to decrypt the packets correctly, anymore.

If all the previous steps are completed successfully, then you finished suc-


Activities
Applications
Wireshark
Feb 28 17:02
Capturing from /tmp/sensniff
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Wireshark · Preferences
Keys

```
No.  Time        Source                     Destination             Protocol      Length Info
   1 0.000000    ::212:4b00:c61:ed03        ::212:4b00:c65:be07     ICMPv6           103 RPL Control (Destination Advertisement Object)
   2 0.000239                                                       IEEE 802.15.4      5 Ack
   3 0.270063    ::212:4b00:c65:be07        ::212:4b00:c61:ed03     ICMPv6            61 RPL Control (Destination Advertisement Object Acknowledgment)
   4 0.270328                                                       IEEE 802.15.4      5 Ack
   5 2.985336    fe80::212:4b00:c61:ed03    ff02::1a                ICMPv6           115 RPL Control (DODAG Information Object)
   6 3.195583    fe80::212:4b00:c65:be07    ff02::1a                ICMPv6           115 RPL Control (DODAG Information Object)
   7 9.555698    fe80::212:4b00:c61:ed03    ff02::1a                ICMPv6           115 RPL Control (DODAG Information Object)
   8 16.906075   fe
   9 50.852647   fe
  10 56.012999   fe                                                                      ation)
  11 58.533406   fe
  12 61.023095   fe
  13 61.098545   fe
  14 61.188664   fe
  15 61.189323
  16 63.888630   fe
  17 63.889205
  18 67.008432   :                                                                       Object)
  19 67.008592
  20 67.308627   :                                                                       Object Acknowledgment)
```

IEEE 802.15.4 Low-Rate Wireless PAN

802.15.4 Ethertype (in hex)	0x809a
FCS format	TI CC24xx metadata
✓ Dissect only good FCS	
✓ Enable ACK tracking	
✓ Assume 802.15.4e-2012 for compa	
Static Addresses	Edit…
Decryption Keys	Edit…
Security Suite (802.15.4-2003)	AES-
☐ Extend authentication data (802.15	

Decryption key	Decryption key index	Key hash
101112131415161718191a1b1c1d1e1f	0	No hash

/root/.config/wireshark/ieee802154_keys

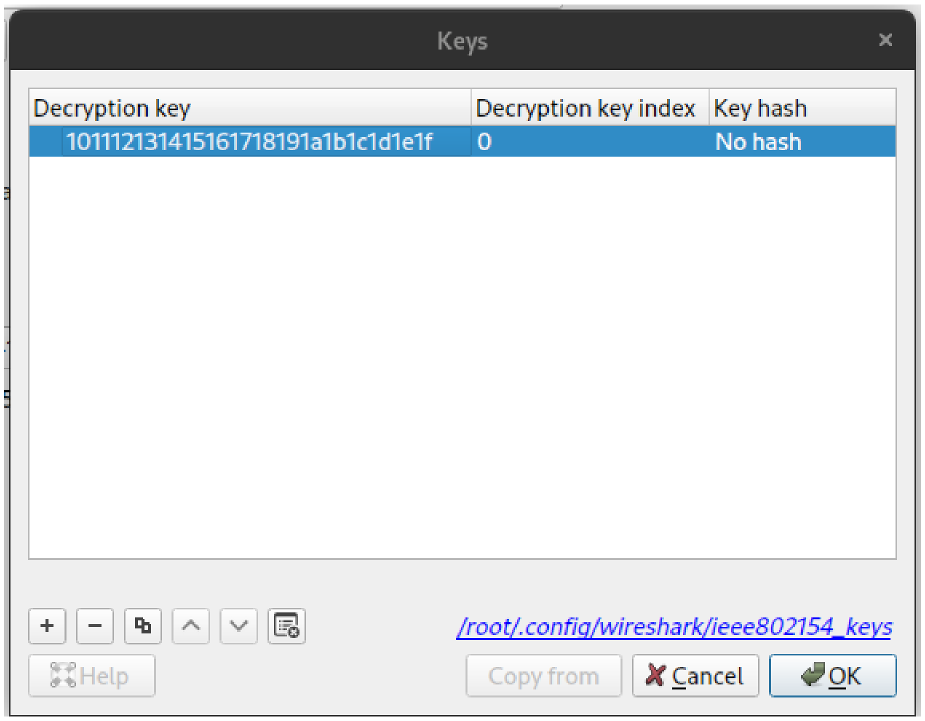
```
    Epoch Time: 16775945
    [Time delta from pre
    [Time delta from pre
    [Time since referenc
    Frame Number: 8
    Frame Length: 115 by
    Capture Length: 115
    [Frame is marked: Fa
    [Frame is ignored: F
    [Protocols in frame:
    [Coloring Rule Name:
    [Coloring Rule Strin
▾ IEEE 802.15.4 Data, Ds
  ▾ Frame Control Field:
      .... .... .... .00
      .... .... .... 1..
      .... .... ...0 ...
      .... .... ..0. ...
      .... .... .1.. ...
      .... .... 0... ...
      .... ...0 .... .... = Sequence Number Suppression: False
      .... ..0. .... .... = Information Elements Present: False
      .... 10.. .... .... = Destination Addressing Mode: Short/16-bit (0x2)
      ..01 .... .... .... = Frame Version: IEEE Std 802.15.4-2006 (1)
      11.. .... .... .... = Source Addressing Mode: Long/64-bit (0x3)
    Sequence Number: 130
    Destination PAN: 0xabce
    Destination: 0xffff
    Extended Source: TexasIns_00:0c:65:be:07 (00:12:4b:00:0c:65:be:07)
  ▾ Auxiliary Security Header
    ▾ Security Control Field: 0x16, Security Level: Encryption with 64-bit Message Integrity Code, Key Identifier Mode: Explicit Key with 4-octet Key Source
        .... .110 = Security Level: Encryption with 64-bit Message Integrity Code (0x6)
        ...1 0... = Key Identifier Mode: Explicit Key with 4-octet Key Source (0x2)
```

```
0000  49 d8 82 ce ab ff ff 07  be 65 0c 00 4b 12
0010  08 00 00 00 00 00 00 00  00 07 02 53 ad d7
0020  7c 90 9c 3d 1c fd 9c ae  9d 33 e2 96 1e 34
0030  dc b8 5c 6f 08 5b 57 03  a7 86 e0 f5 24 30
0040  d2 c1 89 03 7f f6 b9 a3  e2 2d d8 fe 97 9f
0050  50 d0 94 72 b9 99 1b 93  f7 7e db dd b8 71
0060  4c 20 9f e9 4e 30 80 63  13 67 3d 41 cd ed
0070  77 e2 be
```

Frame (115 bytes)   Decrypted IEEE 802.15.4 payload (80 byt

/tmp/sensniff: <live capture in progress>   Packets: 84 · Displayed: 84 (100.0%)   Profile: Default

Figure 8: Configuration of the decryption key.



Decryption key	Decryption key index	Key hash
101112131415161718191a1b1c1d1e1f	0	No hash

/root/.config/wireshark/ieee802154_keys

Figure 9: Decryption Key.

cessfully the second lab session.

You can also have a look around in the code of *contiki-ng* to familiarize with the several components of the protocol stack and the available examples.

References

[1] Contiki-NG, "Contiki wireshark." [Online]. Available: https://docs.contiki-ng.org/en/develop/doc/platforms/cc2538dk.html#build-a-sniffer-live-traffic-capture-with-wireshark