A RFID-based IoT Cybersecurity Lab in Telecommunications Engineering

Santiago Figueroa
Ceit-IK4
Donostia / San Sebastian, Spain
sfigueroa@ceit.es

Javier Añorga
Ceit-IK4, TECNUN
Donostia / San Sebastian, Spain
janorga@ceit.es

Josune Hernantes
TECNUN
Donostia / San Sebastian, Spain
jhernantes@tecnun.es

Juan Francisco Carías
TECNUN
Donostia / San Sebastian, Spain
jfcarias@tecnun.es

Saioa Arrizabalaga
Ceit-IK4, TECNUN
Donostia / San Sebastian, Spain
sarrizabalaga@ceit.es

***Abstract*— The Internet of Things (IoT) is a distributed system of physical objects that requires the integration of hardware (e.g., sensors, actuators) and network communications in order to collect and exchange data. IoT smart objects need to be identified in some way to determine the origin of the data. One of the best-positioned technologies to perform identification is RFID (Radio Frequency Identification), which has gained a lot of popularity in applications like access control, payment cards or logistics. This paper presents a RFID-based IoT Cybersecurity laboratory: it provides a guideline for understanding and analyzing in depth the RFID technology from the cybersecurity point of view. The application is based on Proxmark3, which is a suite of HW/SW tools that facilitates message eavesdropping and emulation of both tags and readers. All the software, firmware and hardware that is described here is open source and open design.**

Keywords*— *RFID, Cybersecurity, IoT, Internet of Things, Telecommunications Engineering

I. INTRODUCTION

This lab is part of the Smart Cities subject in the Master in Telecommunication Engineering at the University of Navarra's School of Engineering (TECNUN) in Donostia, San Sebastian (Spain). The same infrastructure is also part of the IoT Space, an open Internet of Things (IoT) laboratory environment that promotes training and innovation in young researchers (final undergraduate or Master's project), who work on real projects related to the concepts deployed by IoT, which also serves as an experimentation node with a focus on Industry 4.0 applications.

Smart Cities subject (which has 5 ECTS) is part of the Network, Systems and Services module (20 ECTS) of the Telecommunication Engineering plan study. The subject "Smart Cities" includes five sections: *I. Network and Sensors, II. Applications, Services and Protocols, QoS and IP Networks, IV. Security in Smart Cities and V. Visual Analytics.* RFID-based IoT Cybersecurity Lab, will be part of next 2018-2019 academic year as part of the section called *Security in Smart Cities*.

There are other similar initiatives that have previously been carried out also in Tecnun and Ceit-IK4, which are aligned with this new laboratory, such as [1] and [2]. In [1] a new tool for cybersecurity learning is presented; in [2], another PBL (Project-Based Learning) developed for the same subject is described, which is currently part of *Network and Sensors* section. IEEE 802.15.4 2.4GHz WSN standard is used to build a sensor IoT environment to measures parameters as temperature, humidity and light.

RFID-based IoT Cybersecurity Lab was created due to the practical experience based in PBL (Project-Based Learning) received by [2] that will be extended to Cybersecurity branch and will allow to apply innovative student learning methodologies to teach cybersecurity to Master students.

Our use of PBL is designed to improve the efficiency of cybersecurity education and to help students develop the wide range of skills needed to be a cybersecurity professional, including technical aspects, team work, making judgments and developing as lifelong learners.

Rationale for using PBL in Cybersecurity are:

- Reduce 'information overload' and stop students from learning huge amounts of unnecessary theoretical detail.
- Address perceived weakness in traditional 'didactic' teaching methods.
- Improve students' control over their learning by providing more opportunity to be self-directed, to locate what they need to know and give them possession of their learning.
- Improve students' interpersonal skills.

This paper provides a detailed guideline for going through the security analysis of RFID (Radio Frequency Identification) technology based on a PBL methodology, through the cryptographic analysis to RFID Tags, used in popular applications like control access, payment cards and logistics. Although the present work is focused on the labels Mifare Classic (MFCT), it must be taken into account that similar analysis are carried out for cards such as Mifare Ultralight, EM4100, T5577, HID Prox Card II, to name some of the most popular cards in the market.

The overall goal is to improve student learning via a multidisciplinary approach which encourages students to

978-1-5386-0928-6/18/$31.00 ©2018 IEEE

collaborate with their peers, ask questions, develop a deeper understanding of the subject and take actions in solving real-world challenges. Also, students establish essential questions which reflect their interests in information security, formulate challenges on how to safeguard confidential information from cyber-attacks and then came up with solutions to secure their information and network. The pentesting approach through the exploitation of different vulnerabilities is another focus of this Lab.

This paper is structured as follows. Section II shows the objective pursued. Section III describes the material used in the practice lab. Section IV introduces the necessary steps to configure Lab devices, while section V describe how is integrated this devices and applications. Section VI describes the process of capturing and analyzing RFID communications with the Wireshark software tool. Section VII depicts how to use the Proxmark3 security tool in reader mode, whereas section VIII shows how to use the same tool in eavesdropping mode. Section IX details a cryptographic analysis of a Mifare Classic Card and Section X the Proxmark3 as a cryptographic analysis tool. Finally, Section XI provides some conclusions regarding the competences that students should have acquired after finishing this lab.

II. OBJECTIVES

The objective of this paper is to provide a practical introduction to RFID security analysis using the radio frequency identification tool Proxmark3. The outcome of the proposed practice lab is a guide that offers the following information:

- Description of the material to be used in the lab.
- Introduction to the tools with the necessary steps for getting started.
- Practical exercise in connecting a tag with a RFID reader.
- How to use Proxmark3 as a reading, eavesdropping and cryptographic analysis tool.

III. MATERIAL

The material needed to perform the proposed lab includes the following hardware: Proxmark3, a Raspberry Pi 3B, a High Frequency (HF) Reader and RFID Tag. The required software includes: the Proxmark3 client, Minicom (text-based serial port communications program), and Wireshark (network analyzer)[1].

A. *Proxmark3*

Proxmark3[2,3] (PM3) supports both low (125 kHz-134 kHz) and high frequency (13.56MHz) signal processing, which is enabled by two independent parallel antenna circuits. Both antennas are connected to a 4-pin Hirose connector, and in turn it is connected to an external loop antenna [3]. PM3 can be used in reading mode, eavesdropping mode or in card emulation mode. The reading and eavesdropping mode will be analyzed throughout this paper.

The signal from the antenna is routed through the FPGA (Field Programmable Gate Array) after it has been digitized by an 8-bit ADC (Analog-to-Digital Converter). The FPGA relays the information needed to perform the signal decoding to the microcontroller. The core of this microcontroller is an ARM processor that is in charge of the protocol. It receives the digital signal from the FPGA and decodes it. The decoded signal can just be copied to a buffer in the EEPROM (Electrically Erasable Programmable Read-Only Memory).

The PM3 has an USB interface to the computer. The current implementation uses the default Human Interface Device (HID) USB protocol. The microcontroller and the FPGA can be flashed via USB [3].

B. *RaspberryPi3B*

Another important piece of hardware in the Lab is the RaspberryPi3B4 (RBPI3), which is a small single-board computer. The RBPI3 has a Broadcom System on chip, including a 64-bit processor with four ARM Cortex cores running at 1.2 MHz, four USB 2.0, an Ethernet port, and BLE (Bluetooth Low Energy) and Wi-Fi interfaces, among other powerful features [4].

The RBPI3 has been selected as part of the Lab's hardware to compile the PM3 client due to its portability and easy remote access using SSH (Secure Shell) or VNC (Virtual Network Computing).

C. *HF Reader ISO 14443*

A RFID reader is required in order to get captures in Wireshark and to use the PM3 in eavesdropping mode. The NXP MFRC500 RFID reader has been selected due to its easy management via UART (Universal Asynchronous Receiver Transmitter) communication.

D. *Tag*

MFCT has been selected for a reference analysis. This is an EEPROM memory chip which implements a proprietary secure communication algorithm (CRYPTO1).

MFCT's basic operations are: read, write, increment, and decrement. The memory of the tag is divided into 16 sectors. Each sector is further divided into 4 blocks of 16 bytes each. The last block of each sector is called the sector trailer and stores two secret keys ('A' and 'B' keys) and access conditions corresponding to that sector.

To perform an operation on a block, the reader (MFRC500) must be authenticated with the sector containing that block. The access conditions of that sector determine which key, 'A' or 'B', must be used during the authentication stage. More information about MFCT can be found in [5].

E. *Minicom and HF Reader commands*

A Minicom is a text-based serial port communications program. It is used to talk to external RS-232 devices such as

[1] https://www.wireshark.org
[2] https://proxmark3.com
[3] https://proxmark3.com

mobile phones, routers, and serial console ports. Minicom is installed on the RBPI3 Operating System (Raspbian). With Minicom, the user can exchange commands with the HF Reader. Table 1 shows the commands used in this Lab [6].

TABLE 1 MFRC500 COMMON COMMANDS

Common commands	
Command	***Description***
'c'	Continuous read
's'	Select
'l'	Login
'rb'	Read Block
'wb'	Write Block

F. PM3 app and Wireshark

Both the PM3 client and Wireshark are covered in Section IV in terms of installation and configuration.

IV. CONFIGURING LAB DEVICES

This section contains concise and clear instructions for setting up the working environment and a description of the designed lab architecture.

A. Installing and configuring the RBPI3

First, it is necessary that the Raspbian OS is installed on the RBPI3. To do this, Raspbian must be downloaded, and Noobs[4] is used as the booting system. Then, using Etcher[5], the SD Card with OS Raspbian on it can be flashed. By default, the username is 'pi' and the password is 'raspberry'.

To detect the RBPI3 architecture installed, use the following command (C: Command, R: Console Output), and the response below it should be obtained:

```
C: $ sudo uname –a                    (1)
R: Linux raspberrypi 4.4.50-v7+
```

In addition, SSH and VNC can be enabled for remote access the RBPI3. To activate SSH and VNC servers, use the following command as root user:

```
C: $ sudo raspi-config                (2)
```

and set the services in the Interfacing Options.

Once the server is activated, in the case of SSH, it can be accessed by using the following command:

```
C: $ ssh pi@192.168.2.116             (3)
```

from a client terminal, where "192.168.2.116" is the RBPI3 IPv4 address. The VNC client needs a graphical version like Real VNC[6].

[4] Noobs Documentation: https://goo.gl/pk3AOK
[5] https://etcher.io
[6] https://www.realvnc.com

B. Installing and configuring Minicom serial interface

Install the program as root user with:

```
C: $ sudo apt-get install minicom     (4)
```

In order to find the name of the HF Reader, the next command must be typed into the terminal:

```
C: $ dmesg | grep tty                 (5)
```

The output should be similar to:

```
R: [266.973239] usb 1-1.2: FTDI USB Serial Device converter now attached to ttyUSB0
```

In the output shown above, the serial port is ttyUSB0. This port name will be used in Minicom. Next, the following command must be entered in a terminal as root user:

```
C: $ sudo minicom –s                  (6)
```

Next, choose '`Serial port setup`', and then adjust the Serial Device to the corresponding setting parameters, for example:

```
C: A - Serial Device : /dev/ttyS0     (7)
C: E - Bps/Par/Bits : 9600 8N1
C: F – Hardware Flow Control : No
C: G – Software Flow Control : Yes
```

C. Installing and configuring PM3 on RBPI3

To install and configure PM3, each step recommended in [7] must be followed. When the process is finished, this positive response should be obtained:

```
C: $ sudo ./proxmark3 /dev/ttyACM0    (8)
C: proxmark3> hw version              (9)
C: proxmark3> hw tune                 (10)
```

Command 8 allows running PM3 application using PM3 hardware connected across `/dev/ttyACM0`. Command 9 allows knowing PM3 firmware version installed and command 10 allows identifying antenna type installed.

It has been shown that it is possible to set up PM3 on RBPI3 running Raspbian OS, since Raspbian is a Debian-based OS distribution and has a Linux Kernel (4.4.50-v7+) similar to Ubuntu 16.04.

D. Installing and configuring Wireshark

The Wireshark installation is simple. In a RBPI3 terminal use the following command as root:

```
C: $ sudo apt-get install wireshark   (11)
```

next, the command:

```
C: $ sudo modprobe usbmon             (12)
```

enables the module that allows traffic to be captured via USB. Finally, the command:

```
C: $ sudo chmod +x /sys/kernel/debug/usb/usbmon/1t    (13)
```

grants execution permission to USB device.

In addition, the command

```
C: $ lsusb                                    (14)
```

is needed to know the device number and the bus used. The following response should show up when an HF Reader is connected:

```
R: Bus 001 Device 007: ID 0403:6015 Future Technology Devices International, Ltd Bridge (I2C/SPI/UART/FIFO)
```

Then, it is necessary to open Wireshark as root with the command:

```
C: $ sudo wireshark                           (15)
```

V. LAB ARCHITECTURE

Once the materials and the installation for each device have been described, it is important to define the three different configurations that are used in the lab.

The Fig. 1 shows the first configuration used that will be described in the section VI. Similarly, Fig. 2 and Fig. 3 will be described in Section VII and VIII respectively. Each student will have a workstation with devices and open source software. The lab guide indicates which configuration needs to be used thorough the exercises.

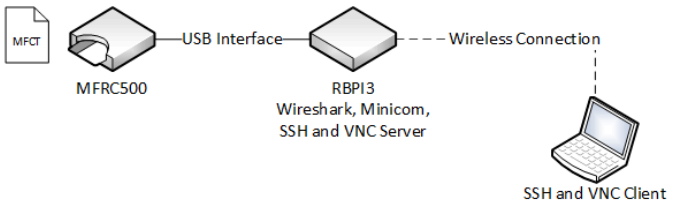

Fig. 1 Capture and analysis with Wireshark mode

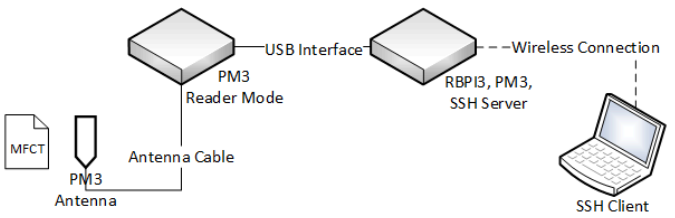

Fig. 2 PM3 in reading mode

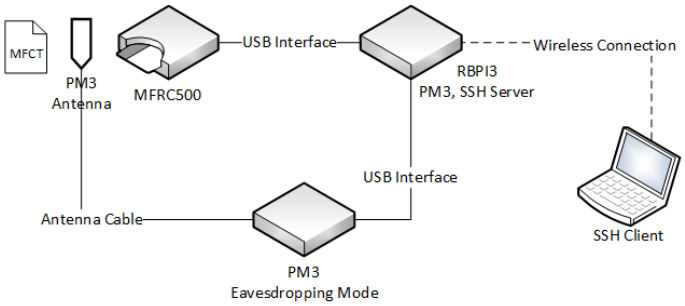

Fig. 3 PM3 in eavesdropping mode

VI. CAPTURING AND ANALYZING WITH WIRESHARK

To perform a security analysis of RFID cards, it is necessary to control the interaction between the reader and the card at low level. For this reason, the following test generates traffic and then performs some captures, so that the interaction can be analyzed and understood.

First, with Wireshark opened and usbmon1 selected, apply the filter `usb.device_address == 7`, which corresponds to the USB device ID for the detected HF Reader. Next, Minicom must be activated so it is possible to have the following captures, as Fig. 4 shows.

Fig. 4 Wireshark capture

Second, in a controlled environment, knowing Mifare Tag parameters like UID and Sector 0 Key A, hold the Mifare Tag to the HF Reader and use the follow commands on Minicom:

TABLE 2 MFRC500 COMMANDS USING MINICOM INTERFACE

Commands Applying	
Command	***Description***
's'	Select the card
'l00AAFFFFFFFFFFF'	Login in sector 0
'rb00'	Read Block 0
'rb01'	Read Block 1
'rb02'	Read Block 2

Now, the Wireshark filter `usb.device_address == 54 & frame.len > 66` must be used to capture a specific data package and leave a control package. Fig. 5 shows the captures before a login process. Fig. 6 and Fig. 7 show the captures after a login process and reading the content of a block.

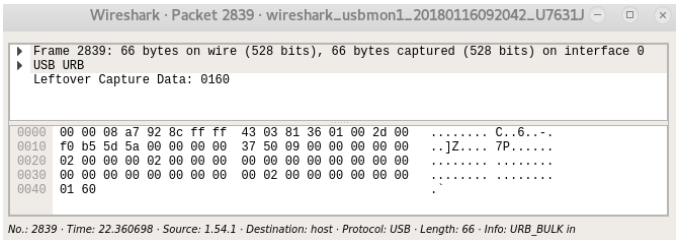

Fig. 5 Wireshark capture before login

In both cases the information which is shown by Wireshark captures do not depict the needed low level information to carry out a security analysis. As Fig. 6 shows, the command

that is sent (`rb00` (Table 2)) is captured, but no more information about the protocol is obtained. For this reason, it is necessary to use PM3.

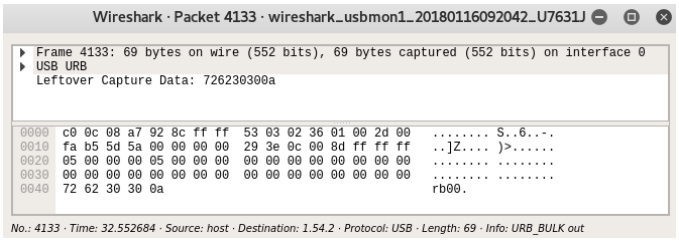

Fig. 6 Wireshark capture after login in block 00 (Part I)

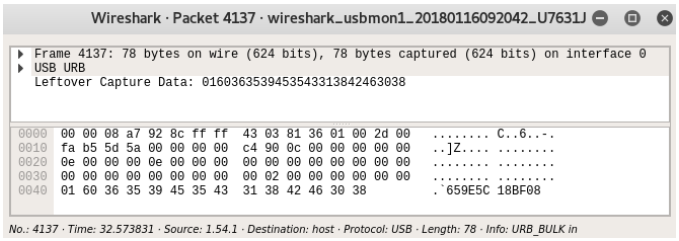

Fig. 7 Wireshark capture after login in block 00 (Part II)

VII. PM3 IN READER MODE

When configured in reader mode, as Fig. 2 shows, PM3 emulates an RFID reader, like MFRC500. The major difference with a real reader is that it allows the execution of commands to know the first interaction between the tag and the reader.

To use the PM3 in Reader mode, a controlled environment is necessary (Fig. 2). Then, the following command can be applied:

```
C: proxmark3> hf search                    (16)
R: UID : 65 9e 5c 18
   ATQA : 00 04
   SAK : 08 [2]
   TYPE : NXP MIFARE CLASSIC 1k | Plus 2k SL1 proprietary non iso14443-4 card found, RATS not supported. No chinese magic backdoor command detected
   Prng detection: WEAK
   Valid ISO14443A Tag Found - Quiting Search
```

Command 16 searches for HF Tag (13.56 MHz), nearby the PM3.

The response provides critical information about the tag (card). For example, the UID Card, [5] in this case, and the low level command exchanged between reader (PM3) and tag, which corresponds to ISO 14443-3 [8].

To study this protocol in depth, execute:

```
C: proxmark3> hf 14a reader                (17)
R: UID : 65 9e 5c 18
   ATQA : 00 04
   SAK : 08 [2]
   Field dropped
```

Command 17 allows acting like an ISO14443 Type A reader.

Details about ISO 14443-3, such as reader pulling, card detection and anti-collision algorithms, are described in [9], [10], [11].

VIII. PM3 IN EAVESDROPPING MODE

While the PM3 Reader Mode allows the study of RFID Cards, the Eavesdropping Mode allows a focus on security aspects, because as shown in Fig. 3, the PM3 antenna receives interaction MFCT – MFRC500.

The following command starts an eavesdropping:

```
C: proxmark3> hf 14a snoop                 (18)
```

Command 18 allows eavesdropping ISO 14443 Type A card.

The interaction that happens in an ISO/IEC 14443 type A communication is described in [12].

It is necessary to hold the PM3 antenna next to the reader and present the MFCT. Blinking lights indicate that the transmission has been captured. If the button on the PM3 is pressed, the flow frames stops. Another way is to wait until the buffer is full. The trace most likely contains more than only the authentication information. Before the reader can exchange messages with a MFCT, it needs to perform the initial communication and the anti-collision protocol [8]. To retrieve the eavesdropped trace from the PM3, the following command 19 can be used:

```
C: proxmark3> hf list 14a                  (19)
```

Fig. 8 represents an example of capture in eavesdropping mode. In this case, a deep interaction between MFCT and MFRC500 is detected. Fig. 9 explains this interaction.

```
proxmark3> hf list 14a
Waiting for a response from the proxmark...
#db# COMMAND FINISHED
#db# maxDataLen=4, Uart.state=0, Uart.len=0
#db# traceLen=815, Uart.output[0]=00000093
Recorded Activity (TraceLen = 815 bytes)
Start = Start of Start Bit, End = End of last modulation. Src = Source of Transfer
iso14443a - All times are in carrier periods (1/13.56Mhz)
iClass    - Timings are not as accurate
      Start |        End | Src | Data (! denotes parity error)      | CRC | Annotation |
------------+------------+-----+------------------------------------+-----+------------+
          0 |   22359968 | Rdr | 26                                 |     | REQA       |
    3725888 |   22363524 | Tag | 04  00                             |     |            |
    7453184 |   22369184 | Rdr | 93  20                             |     | ANTICOLL   |
   11179456 |   22376196 | Tag | 3e  b4  e2  49  21                 |     |            |
   14905408 |   22583136 | Rdr | 93  70  3e  b4  e2  49  21  69  62 |  ok | SELECT_UID |
   18632224 |   22587828 | Tag | 08  b6  dd                         |     |            |
   18634452 |   22610640 | Rdr | 60  02  e7  58                     |  ok | AUTH-A(2)  |
   22358912 |   22620084 | Tag | 24  5c  db  ee                     |     |            |
   22361156 |   22630816 | Rdr |58! 47!  67 bf!  68  6f 4f! b9!     | !crc| ?          |
   22366720 |   22636724 | Tag |60! 21! b9! 3d!                     |     |            |
```

Fig. 8 Eavesdropping mode capture

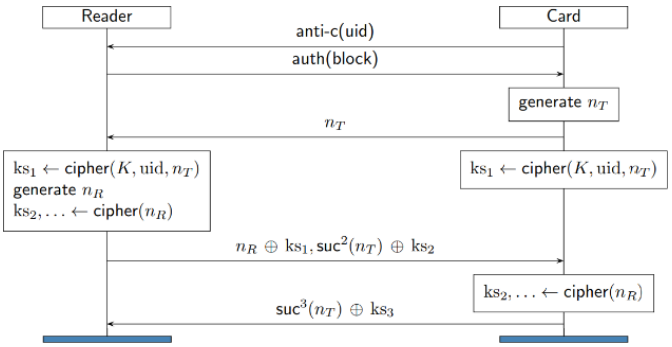

Fig. 9 Interaction between MFCT and MFCR500

The capture will be used in the next section to perform a deep cryptography analysis, which is important for recovering certain information: `uid: 3eb4e249`, `nt: 245cdbee`, `{nr}: 584767bf`, `{ar}: 686f4fb9`, `{at}: 6021b93d`; where:

`uid: Identification number`

`nt: nonce tag`

`nr: nonce reader`

`at: answer tag`

`ar: answer reader`

`{nr}: nonce reader cipher`

`{at}: answer tag cipher`

`{ar}: answer reader cipher`

`suc: successor`

This process is known as "Three pass authentication", which is described by [13] and mentioned below:

- The tag picks a challenge nonce 'nt' and sends it to the reader in the clear.
- The reader sends its own challenge nonce 'nr' together with the answer 'ar' to the tag's challenge.
- The tag finishes authentication by replying at to the challenge of the reader.

The two next sections propose a methodology to develop a cryptography analysis, including a pseudocode and the implementation with PM3.

IX. Cryptography Analysis

The Authentication Protocol is the starting point for the exchange of the ciphered information. This section will analyze the three-pass authentication step by step.

A. Pseudo Random Number Generator (PRNG) analysis side Tag

First, once selected the card and exchanged the '`uid`' with the reader, the tag sends '`nt`' (`245CDBEE` in Fig. 8). The '`nt`' is generated by PRNG as shown in Fig. 10.

According to [14], the random numbers are generated by a linear feedback shift register. The register is 32 bits long, as '`nt`' and the feedback only use the 16 lower bits, having 2^{16}-1 combinations:

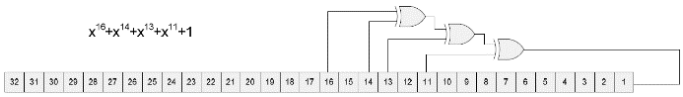

Fig. 10 PRNG scheme

The clock speed and initialization are other important parameters mentioned [14]. In this way, in each oscillation of the clock a new 32-bit number will be generated in the PNRG and therefore a nonce successor e.g: 'suc_2`(nt)`', 'suc_3`(nt)`', ..., as shown Fig. 9.

According to [15], PRNG is initialized at 0xAAAA and clocked every 128 oscillations of the carrier frequency. A frequency of f_{PRNG} = `106 Hz` is given, and the whole search space will repeat every `T=0,618619s`.

Next, a PRNG implementation described by pseudocode enables the sucesor nonce to be calculated, along with the distance between two nonces:

```
01. Load SrcNonce
02. Load TargNonce
03. Distance = 0
04. Rand = SrcNonce
05: Reverse bits in each byte of Rand
06: Reverse in each byte of TargNonce
07: WHILE Rand != Targnonce
   08: Shift Rand left by 1 bit
   09: bit15 XOR bit13 XOR bit12 XOR bit10 of
Rand
   10: Rand = result of (09) OR Rand
   11: Increment Distance by 1
      12: IF Distance > 2^16
      13: THEN
      14: Distance = 0;
15. Exit Loop
16. END IF
18. END WHILE
```

In the next section a PM3 tool with a similar implementation is used to obtain 'suc^2`(nt)`' and 'suc^3`(nt)`'.

B. CRYPTO1 cipher and PRNG analysis reader side

Second, as shown in Fig. 11, when the reader receives the '`nt`' the cipher system is activated, which allows to generate '`ks1`' to obtain '`{nr}`' and to generate '`ks2`' to obtain '`{ar}`'; on the tag side, '`ks3`' is generated to obtain '`{at}`'. Then, it will be used to get '`ks4`', '`ks5`', and so on, thus encrypting the plain text. Both, tag and reader clock are synchronized.

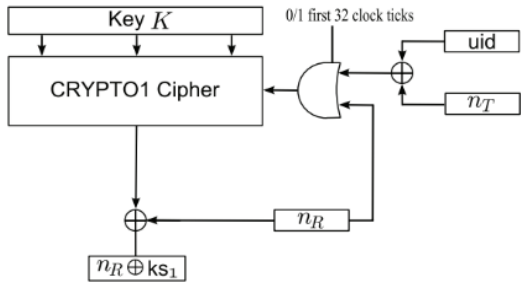

Fig. 11 Initialization diagram

The system provides mutual authentication and data secrecy by means of the so called CRYPTO1 cipher [16].

A CRYPTO1 is a fast cipher based on a 48-bit Linear Feedback Shift Register (LFSR) from which 20 bits are entered into a Boolean equation filter to obtain a bit of keystream. To obtain the next keystream bit, the LFSR is shifted left and filtered.

At every clock tick, the register is shifted one bit to the left. The LFSR has an input bit that is XOR-ed with the feedback bit and then fed into the LFSR to the right.

According to [17], the keystream generated in the first 32 clock ticks is ignored, as shown in Fig. 12, and therefore it is not used to cypher plaintext. This keystream only uses '`K`' (secret key of a sector) and the '`XOR`' between '`nt`' and '`uid`' as feedback bits.

`XOR Representation: nt ⊕ uid ⊕ K`

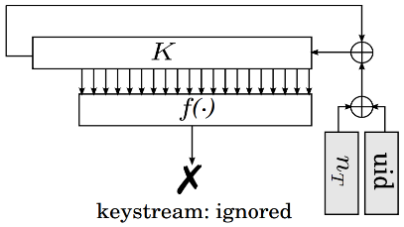

Fig. 12 First 32 clock ticks

Next, the keystream (**'ks1'**) is generated in the second 32 clock (64 clock ticks) ticks and is used to cypher the **'nr'** that PRNG generated (Fig. 13). In this case, **'{nr}'** is obtained. This keystream uses:

`XOR Representation: nt ⊕ (uid ⊕ nt) ⊕ K ⊕ 'feedback bits'`

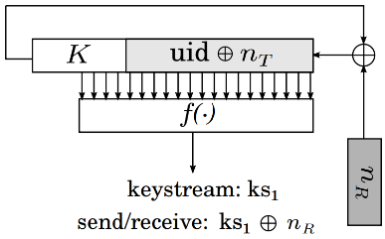

Fig. 13 Second 32 clock ticks

Before the reader can send the response to '`nt`' it is necessary to generate '`ks2`'. Fig. 14 shows the process:

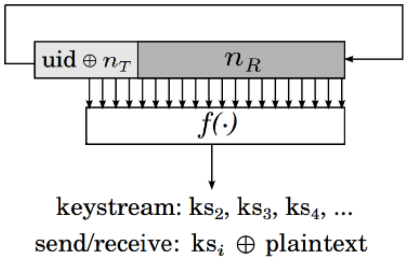

Fig. 14 Runnig mode

For each new '`ks`', a new '`nr`' is generated by the PRNG and used.

`nr ⊕ ks1 = {nr}`

`ar = suc²(nt)`

`suc² (nt) ⊕ ks2 = {ar}`

`nonce and answer reader ciphered: {nr}{ar}`

In Fig. 8, the reader message was `58!47!67BF!686F4F!B9!`, divided in:

`{nr}:58!47!67BF!`

`{ar}:686F4F!B9!`

At this point, the tag is able to update the cipher state in the same way and verify the authenticity of the reader. The remainder of the keystream 'ks3', 'ks4' . . . is now determined and from now on all communication is encrypted, i.e., XOR-ed with the keystream.

C. CRYPTO1 Cipher and PRNG analysis Tag side

The Tag side has the same cryptographic system as shown in Fig. 11. Synchronization between clocks will allow the tag (once the reader's message (`{nr}{ar}`) is received) generating the answer tag cipher:

`at = suc³(nt)`

`suc³(nt) ⊕ ks3 = {at}`

The keystream 'ks3' is generated with the process shown in Fig. 14.

X. CRYPTOGRAPHY ANALYSIS USING PM3

PM3 includes an important tool that allows the nonce and the keystream to be calculated. Then, using the same values as it is given in Fig. 8, the command and the response are shown.

```
C: /mfkey> sudo ./mfkey64 3eb4e249 245cdbee
584767bf 686f4fb9 6021b93d                    (20)
```

Recovering key for:

```
R: uid: 3eb4e249
   nt: 245cdbee
```

```
{nr}: 584767bf
{ar}: 686f4fb9
{at}: 6021b93d
LFSR successors of the tag challenge:
ar: b095c7be
at: 5843ff81
Time spent in lfsr_recovery64(): 0.12 seconds
Keystream used to generate {ar} and {at}:
ks2: d8fa8807
ks3: 386246bc
```

Although '`ks2`' and '`ks3`' are used to obtain the values of '`{ar}`' and '`{at}`' respectively, the following process is done to prove that the values of '`ks2`' and '`ks3`' recovered by the PM3 are the correct ones.

$suc^2(nt) \oplus ks2 = 0x686f4fb9$

$suc^2(nt) = 0xb095c7be \oplus$

$ks2 = 0xd8fa8807$

$suc^3(nt) \oplus ks3 = 0x6021b93d$

$suc^3(nt) = 0x5843ff81 \oplus$

$ks3 = 0x386246bc$

Fig. 15 shows how ks2 and ks3 have been calculated respectively through the XOR operation.

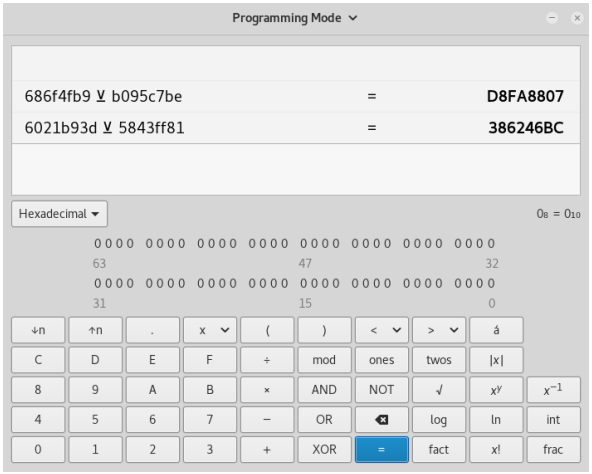

Fig. 15 Xor testing

XI. Conclusions

The Smart Cities subject within the Networks, Systems and Services module in the Master's in Telecommunications Engineering at Tecnun (University of Navarra) and the IoT Space aims to provide the students with a deep understanding of what Smart Cities are and competences they will need. As part of the skills needed to develop services and applications for Smart Cities, RFID Cybersecurity has been found to be essential. This paper has provided a guide for students that uses the PBL methodology. After following the instructions and completing the guide, students should be able to perform an in-depth cybersecurity analysis of the RFID technology, using a powerful tool like Proxmark3 and other relevant tools like Wireshark, Minicom and Raspberry Pi. Currently more PBL labs are being developed in order to develop skills related to other RFID technologies as well Bluetooth Low Energy (BLE), Zigbee and Wireless Hart technologies, all of which are inherent to any Smart Cities application.

References

[1] S. Alonso-Arce, Maykel; Añorga, Javier; Arrizabalaga, "Cybersecurity Challenge Learning Tool (CCLT): A new way to learn Cybersecurity," *III Jornadas Nac. Investig. en Ciberseguridad*, pp. 166–167, 2017.

[2] M. Alonso-Arce, J. Añorga, S. Arrizabalaga, and P. Bustamante, "A wireless sensor network PBL lab for the master in telecommunications engineering," *Proc. 2016 Technol. Appl. to Electron. Teaching, TAEE 2016*, 2016.

[3] F. D. Garcia, G. D. K. Gans, and R. Verdult, "Tutorial : Proxmark , the Swiss Army Knife for RFID Security Research," pp. 1–9, 2012.

[4] E. H. M. Les Pounder, Mayank Sharma, Jonni Bidwell, Neil Mohr, Neil Bothwick, *The Ultimate RaspberyPi Handbook*. Future, 2016.

[5] B. V Nxp, "MF1S70YYX_V1: MIFARE Classic EV1 4K - Mainstream contactless smart card IC for fast and easy solution development," no. November, pp. 1–37, 2017.

[6] U. Manual, "HF MIFARE Easy Module," *Middle East*, vol. 43, no. 0, pp. 1–63, 2008.

[7] Iceman, "Kali Linux," 2017. [Online]. Available: https://github.com/Proxmark/proxmark3/wiki/Ubuntu Linux.

[8] "ISO/IEC 14443-3:2011," 2016. [Online]. Available: https://www.iso.org/standard/50942.html.

[9] R. J. Rodríguez and V. Pepe, "On Relaying NFC Payment Transactions using Android device," in */Rooted*, 2015.

[10] J. Vila and R. J. Rodríguez, "Experiences on NFC Relay Attacks with Android: Virtual Pickpocketing Revisited," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2015, vol. 9440, pp. 87–103.

[11] R. J. Rodríguez and V. Pepe, "Practical Experiences on NFC Relay Attacks with Android: Virtual Pickpocketing Revisited," 2015.

[12] T. Instruments, "ISO/IEC 14443 Overview," 2014.

[13] ISO, "INTERNATIONAL STANDARD ISO/IEC 9798-2," vol. 1999, 1999.

[14] K. Nohl and D. Evans, "Reverse-Engineering a Cryptographic RFID Tag," *Science (80-.).*, no. July, pp. 185–193, 2008.

[15] K. E. Penri-Williams, *Implementing an RFID MIFARE CLASSIC Attack*. 2009.

[16] F. D. Garcia *et al.*, "Dismantling MIFARE classic," in *European Symposium on Research in Computer Security*, 2008, pp. 97–114.

[17] G. de K. Gans, "Outsmarting Smart Cards," Radboud Universiteit Nijmegen, 2013.