Physical-layer Authentication in WiFi Networks
Final Report

Group Members
Liangcheng Yu, Karl Svantorp
Xavi Casas, Baoqing She

EQ2440 Project in Wireless Communication
Advisors: Henrik Forssell, Ragnar Thobaben, Lars Kildehøj
Mats Bengtsson, Baptiste Cavarec

May 22, 2017

Abstract

Nowadays, the traffic demand is growing exponentially and, as a consequence, the networks have been redesigned in order to support the new emerging technologies, like Internet of Things (IoT), 4K processing and virtual reality.
In the context of IoT and Machine-to-machine (M2M) Communications, several devices are connected to the wireless network, sharing information continuously. Most of the related applications require confidential communication links, and therefore network security has become an emerging field in the current research.
Authentication is a concept within the field of security, which relies on the ability to verify the identity of, for instance, a device that is sharing confidential information. While several algorithms are implemented on higher levels of the network, the aim of our project is to develop physical-layer authentication mechanisms.
Physical-layer authentication consists on the exploitation of the physical attributes (features) of either the reciprocal channel or the analog front-end (AFE) imperfections of the transmitting and receiving devices.
We have analyzed a combination algorithm to address the fact that these features have case dependent limitations and advantages. A simplified version has been implemented in real-time, with results showing an accuracy higher than 95%.

Contents

1 Introduction

Authentication is a concept within the field of security defined as the ability to verify the identity of a transmitting device. As the global communication system evolves, wireless devices become an indispensable part of our everyday life. Since the traffic demand is growing exponentially, the networks have been redesigned, thus increasing forms of intrusion, and that makes conventional cryptographic techniques more vulnerable to attacks. Therefore, it is desirable to develop new mechanisms.

One can find several existing techniques in the literature, often implemented using key-based cryptography in higher layers of the network [1], [2]. However, those techniques introduce latency to the communication scheme. Also, the key generation and detection add security overhead into the transmitted frames. Furthermore, the Media Access Control (MAC) address of a device is very easy to modify, making it a vulnerability in the existing techniques.

PHY-Authentication exploits physical attributes in the hardware and communication link, called features, to decide whether a transmitter is legitimate. This approach increases the resiliency and robustness of the authentication algorithm [3]. The reciprocal properties of the channel between a transmitter and a receiver, and the analog front-end (AFE) imperfections of wireless devices constitute two kinds of physical-layer features for authentication (Section. 2 in this document).

In this project we investigate the feasibility and accuracy of a physical layer authentication scheme. By measuring the features Signal-to-Noise Ratio (SNR), Carrier Frequency Offset (CFO) and the channel estimation, we develop a simple authentication scheme with a precision of at least 95%. This is first analyzed in MATLAB, and later implemented as a real-time solution. To achieve this we use the software GNU Radio, which is a highly modular software for signal processing and software defined radios (SDR) in Linux. The SDRs we use, called Universal Software Radio Peripheral (USRP), are unlike other regular consumer radios highly adaptable and offer good performance for a variety of different setups.

2 Theoretical Simulations

Physical-layer authentication complements higher-level cryptography techniques by exploiting both the channel-based and device-specific features of the radio link.

Analog Front-End (AFE) imperfection-based features, such as In-phase/Quadrature Imbalance (IQI) [6], Carrier Frequency Offset (CFO) [7] among others, are specific to different devices, thus being utilized to differentiate between a set of transmitters. However, the difference between devices can be rather small, especially when the noise dominates. Also, they are easy to fake in some cases.

The frequency response of the wireless channel is highly varying in time and depends on the specific environment. For example, the obstacles between the transmitter and receiver change the multi-path components of the received signal.

Since both approaches have disadvantages, feature combination methods can be used to increase the robustness of the authentication algorithms. In [17], a weighted combination of multiple features is used as a high-performance algorithm for physical layer authentication.

In this section, we first introduce some features of each classification and analyze their performance using MATLAB. Then, a multiple feature combining method is implemented.

In order to evaluate the performance of our algorithms, we use the probabilities of False Alarm (P_{FA}) and Miss Detection (P_{MD}) as defined in detection theory. Also, the terminology of Alice, Bob and Eve is introduced, which is commonly used in the literature. Alice is the legitimate transmitter, while Eve is the spoofing entity who tries to mimic Alice. Bob is the receiver, who is supposed to authenticate whether Alice is transmitting. In this context, we define P_{FA} as the probability of detecting Eve when Alice is transmitting and P_{MD} as identifying Alice when Eve transmits.

2.1 Carrier Frequency Offset (CFO)

In this subsection, we focus on authentication using CFO, which is applied from [7].

1. **Modeling of time-invariant CFO**
   Fig. 1 demonstrates a diagram of physical layer authentication using CFO. At the transmitter, the training sequence of length L_s is repeated N_s times. Hence, the total length of the resulting training sequence is $L_t = L_s * N_s$. Therefore, the received signal without CFO satisfies $y(n) = y(n + L_s)$.

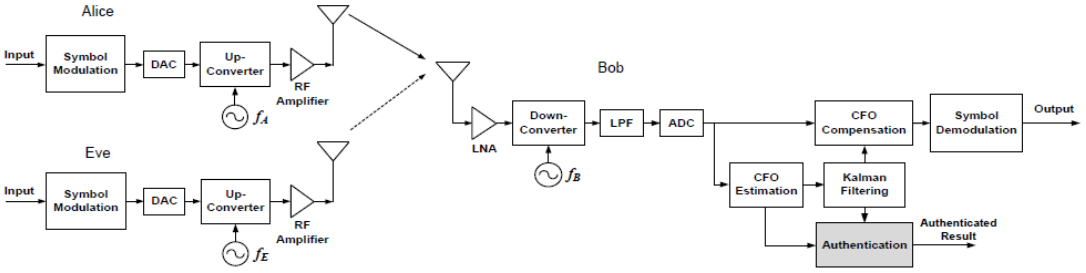

Figure 1: Block diagram for CFO estimation and correction

The corresponding signal with CFO, considering an AWGN channel, is given below:

$$y(n) = e^{j2\pi\epsilon n} \sum_{l=0}^{L-1} h(n-l)x(n) + w(n), \tag{1}$$

where $x(n)$ is the transmitting signal, $h(l)$ are the channel coefficients and ϵ is the normalized CFO. $w(n)$ is zero-mean complex Gaussian noise.

The CFO is obtained every data frame periodically and then normalized by the sampling frequency. The received CFO of Eve ($\Delta f = f_B - f_E$) and Alice ($\Delta f = f_B - f_A$) are different considering the invariant case. First, for each received frame Δf is calculated. Then, the authentication can be achieved by using a binary hypothesis test.

$$\begin{cases} H_0 : \Delta f = \Delta f_A \\ H_1 : \Delta f \neq \Delta f_A \end{cases}, \tag{2}$$

If the CFO estimates $\Delta\hat{f}$ is consistent with Δf_A, H_0 is accepted. Details will be discussed in the following section.

2. **Device authentication method**

The authentication method follows two steps:

- **CFO estimation**

  Assuming the communication system described above, the transmitting signal is periodic with length L_s. As a result, CFO can be estimated by self-correlating the received signal.

$$\hat{\epsilon} = \frac{1}{2\pi L_s}\tan^{-1}\{\frac{S(\sum_{n=0}^{L_t-1} y^*[n]y[n-L_t])}{R(\sum_{n=0}^{L_t-1} y^*[n]y[n-L_t])}\}, \tag{3}$$

Where $\tan^{-1}$ is the inverse tangent function, R( ) and S( ) represent the real and imaginary parts respectively. Here, $\hat{\epsilon}$ is within the range $\hat{\epsilon} \in [\frac{-1}{2L_t}, \frac{1}{2L_t}]$. At high SNR, $\hat{\epsilon}$ is an unbiased estimate of ϵ, with $E(\hat{\epsilon} - \epsilon) = 0$ and variance $\sigma_\epsilon^2 = E[(\hat{\epsilon} - \epsilon)^2] = \frac{\sigma_n^2}{4\pi^2 L_t^3 \sigma_n^2}$. Therefore, CFO estimate can be viewed approximately as a random Gaussian variable with the true CFO ϵ as its mean and with variance σ_ϵ^2.

$$\hat{\epsilon} = \epsilon + w_\epsilon, \tag{4}$$

- **Authentication scheme**

Based on the description above, the authentication scheme can be formulated as the following binary hypothesis test:

$$\begin{cases} H_0 : \hat{\epsilon} = \epsilon_A + w_\epsilon \\ H_1 : \hat{\epsilon} = \epsilon_A + w_\epsilon \quad , \end{cases} \tag{5}$$

where ϵ_A, ϵ_E are the CFO related to Alice and Eve respectively. Therefore, authentication can be simply achieved by comparing the current CFO estimate $\hat{\epsilon}$ with ϵ_A.

$$|\hat{\epsilon} - \epsilon_A| \underset{H_0}{\overset{H_1}{\gtrless}} T. \tag{6}$$

After computation [7], The decision threshold T is obtained from $T = \sigma_\epsilon Q^{-1}(\frac{P_{FA}}{2})$, where σ_ϵ is the theoretical value of the CFO estimation error $\frac{\sigma_n^2}{4\pi^2 L_T^3 \sigma_n^2}$ However, since this approximation applies only at high SNR, T mismatches P_{FA} at low SNR.

2.2 In-phase/Quadrature Imbalance (IQI)

In this subsection, we focus on authentication using IQI, which is applied from paper [6].

1. **Modeling of IQI**

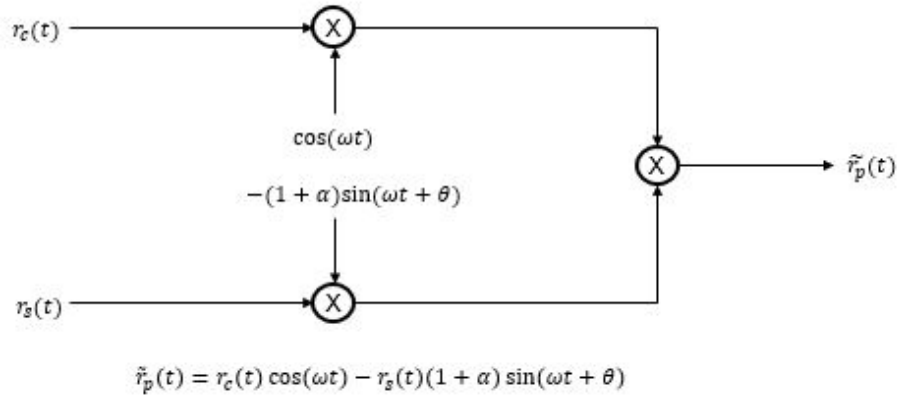

Figure 2: Model of In-phase/Quadrature Imbalance(IQI)

We assume the I branch to be ideal and the Q branch is modeled with IQI (i.e., asymmetric imbalance), as showed in Fig. 2.

When the baseband signal is up-converted into passband signal $\hat{r}_p(t)$, gain imbalance α and phase shift imbalance θ are introduced due to the imperfection of the transmitter. The corresponding baseband equivalent of the signal, $\hat{r}_{bb}(t)$, is:

$$\hat{r}_{bb}(t) = \hat{r}_c(t) + j(1+\alpha)\hat{r}_s(t)e^{j\theta}, \tag{7}$$

Representing $\hat{r}_{bb}(t)$ using r(t), we get:

$$\hat{r}_{bb}(t) = r(t)\mu + r^*(t)\nu, \tag{8}$$

where, $\mu = \frac{1}{2}[1 + (1+\alpha)e^{j\theta}$, $\nu = \frac{1}{2}[1 - (1+\alpha)e^{j\theta}$.

The received signal is represented as:

$$y(n) = (\mu r(n) + \nu r^*(n))h(n) + w(n), \tag{9}$$

Accordingly, we assume our measurement at receiver consists of N observations, then we get:

$$f = (m + w_m) + j(k + w_k), \tag{10}$$

where the N $\times$ 1 vector f denotes N estimated IQI Imbalance based on observations. A comparison between the received fingerprint and validated fingerprint is required to figure out whether Alice is transmitting. We assume the fingerprint of Alice is:

$$f_0 = m_0 + jk_0. \tag{11}$$

Hence, after getting IQI estimate, it is required to compare it with f_0 as:

$$f - f_0 = \Delta m + w_m + j(\Delta k + w_k) = c + jd, \tag{12}$$

where, $c = R\{f - f_0\}$, $d = I\{f - f_0\}$, $\Delta m = m - m_0$, $k = k - k_0$, and w is estimate noise. Finally, binary hypnosis testing is used to decide whether the received signal is from legitimate transmitter.

$$\begin{cases} H_0 : \Delta m_{[i]} = \Delta k_{[i]} = 0 \\ H_1 : {\Delta m_{[i]}}^2 + {\Delta k_{[i]}}^2 \neq 0 \end{cases}, \tag{13}$$

2. Device authentication method

In this section, device authentication based on hypothesis testing is demonstrated. To achieve the goal, we use the likelihood ratio test (LRT), which is able to maximize P_D within a required P_{FA}. three 2N $\times$ 1 vectors are given:

$$a = [c_{[1]}d_{[1]}c_{[2]}d_{[2]}...c_{[N]}d_{[N]}]^T \tag{14}$$

$$b = [\Delta m_{[1]}\Delta k_{[1]}\Delta m_{[2]}\Delta k_{[2]}...\Delta m_{[N]}\Delta k_{[N]}]^T \tag{15}$$

$$c = [w_{m[1]}w_{k[1]}w_{m[2]}w_{k[2]}...w_{m[N]}w_{k[N]}]^T. \tag{16}$$

Then, we obtain:

$$a = b + w. \tag{17}$$

In this case, the hypothesis testing model is equivalent to:

$$\begin{cases} H_0 : b = 0 \\ H_1 : b \neq 0 \quad . \end{cases} \tag{18}$$

The likelihood function of a is given:

$$p(a; b) = \frac{1}{2\pi\theta^2} exp[\frac{-(a-b)^T(a-b)}{2\theta^2}], \tag{19}$$

If the inequation below is satisfied, hypothesis H_1 is made.

$$G(a; b) = \frac{p(a; b_{H_1})}{p(a; b_{H_0})} > T, \tag{20}$$

For simplicity, detailed mathematical processes [6] are not included in this report. After some computation, we can conclude that:

$$A = \frac{a^T a}{\sigma^2} \begin{cases} \chi^2_{2N}, & \text{under } H_0 \\ \chi^2_{2N}(\rho), & \text{under } H_1, \end{cases} \tag{21}$$

where χ^2_{2N} and $\chi^2_{2N}(\rho)$ are the central chi-squared distribution and non-central chi-squared distribution with 2N degrees of freedom respectively. Hence, the false alarm rate is calculated as:

$$P_{FA} = P\{A > T|H_0\} = Q_{\chi^2_{2N}}(T), \tag{22}$$

where $Q_{\chi^2_{2N}}()$ is the right-tail probability for χ^2_{2N} random process. Rearranging the formulation above, threshold T can be represented by P_{FA} as:

$$T = Q^{-1}_{\chi^2_{2N}}(P_{FA}). \tag{23}$$

The P_D is given as:

$$P_D = \int_T^{-\infty} p_{H_1(A)} dA. \tag{24}$$

2.3 Invariant Channel

In this subsection, we focus on authentication using channel information, which is applied from [5].

As mentioned before, different locations of the transmitter will induce different scatters. Fig. 3 shows a typical multi-path environment between entities. The paths between the transmitter and receiver vary when the entity moves or the environment changes.

In this project, we assume a static multi-path environment where both entities are fixed (i.e., invariant channel).

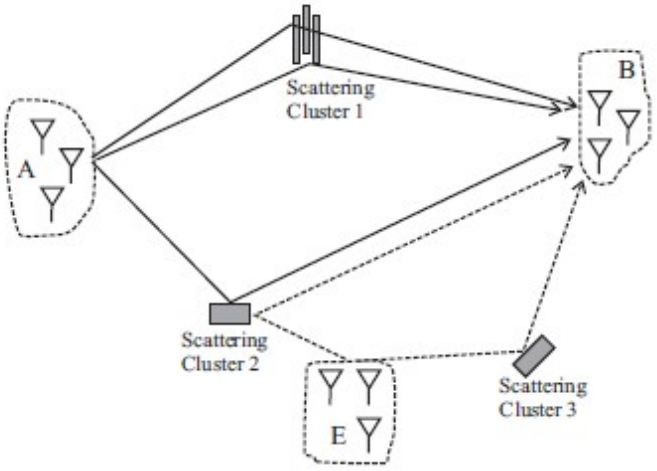

Figure 3: Simulated SNRs vs P_{MD} with different false alarm rate P_{FA}

In our model, we assume Bob stores information about the channel when Alice transmits. Bob obtains a noisy estimate of the channel response between Alice and him, $\hat{H}_{AB}$. After storing the true value of the legitimate channel, Bob starts receiving samples from an unknown transmitter with channel response $\hat{H}_t$.

$$\hat{H}_{AB} = H_{AB}e_1^{j\phi} + N_1, \tag{25}$$

$$\hat{H}_{AB} = H_t e_2^{j\phi} + N_2, \tag{26}$$

where N_1 and N_2 are i.i.d complex Gaussian noise samples $CN(0, \sigma^2)$. Then, it is required that Bob compares the new channel estimate with the validated one to

decide who is the transmitter.

$$\begin{cases} H_0 : H_t = H_{AB} \\ H_1 : H_t \neq H_{AB} \end{cases}, \tag{27}$$

To achieve this objective, Bob accepts this hypothesis if the test statistic he computes L is below some threshold.

$$L = \min_{\phi} \frac{1}{\sigma^2} \sum_{i=1}^{M} |\hat{H}_{tm} - \hat{H}_{AB} e^{j\theta}|^2. \tag{28}$$

To obtain L, minimizing ϕ is required. Result is given below:

$$\phi^* = Arg(\sum_{i=1}^{M} \hat{H}_{tm} \hat{H}^*_{AB}). \tag{29}$$

For the high-SNR conditions, we assume $\hat{H}_t(f) = H_t(f)$, $\hat{H}_{AB}(f) = H_{AB}(f)$, when ϕ takes the minimized value ϕ^*. After some computation, we get the distribution of L. For Alice, the test statistic L is a chi square random variable with 2M degrees of freedom:

$$L = \frac{1}{\sigma^2}(\sum_{i=1}^{M} n_{rm}^2 + \sum_{i=1}^{M} n_{im}^2) \sim \chi^2_{2M,0}, \tag{30}$$

For Eve, L becomes a non-central chi-square variable with a non-centrality parameter μ_L:

$$L = \frac{1}{\sigma^2}(\sum_{i=1}^{M} (\Delta h_{im}^* + n_{rm}^2) + \sum_{i=1}^{M} (\Delta h_{im}^* + n_{im}^2)) \sim \chi^2_{2M,\mu_L}, \tag{31}$$

Therefore, the probability of false alarm is given by:

$$P_{FA} = P_{H_0}(L > k) = 1 - F_{\chi^2_{2M}}(k). \tag{32}$$

2.4 Multiple Weighted Characteristics

1. **Modeling**: A multi-characteristics observation is given by Y below. X_r and W denote the actual values of time-invariant characteristics and additive white Gaussian noises (AWGN) respectively. The nth row contains realizations of the nth feature, and is modeled as $y_n \sim N(x_{rn}, \sigma_n^2)$. The matrix Y consists of M realizations of the N features.

$$Y = X_r + W = \begin{pmatrix} x_{r1} & x_{r1} & \dots x_{r1} \\ x_{r2} & x_{r2} & \dots x_{r2} \\ \vdots & \vdots & \ddots \\ x_{rN} & x_{rN} & \dots x_{rN} \end{pmatrix} + \begin{pmatrix} w_{11} & w_{12} & \dots w_{1M} \\ w_{21} & w_{22} & \dots w_{2M} \\ \vdots & \vdots & \ddots \\ w_{N1} & w_{N2} & \dots w_{NM} \end{pmatrix}, \quad (33)$$

Then, subtracting validated characteristics X_v (Alice) from the received ones, we get ΔX.

$$\Delta X = X_r - X_v + W = \begin{pmatrix} x_1 & x_1 & \dots x_1 \\ x_2 & x_2 & \dots x_2 \\ \vdots & \vdots & \ddots \\ x_N & x_N & \dots x_N \end{pmatrix} + W, \quad (34)$$

$\Delta X = 0$ means the current device has the same characteristics as the validated device. therefore, the hypothesis is shown:

$$\begin{cases} H_0 : x_i = 0 \\ H_1 : x_i \neq 0 \end{cases}. \quad (35)$$

2. **Method**: This section describes an authentication algorithm using the results described before. Different weights are assigned to attributes. Then, different devices are identified by using hypothesis testing based on the weighted features. Firstly, we calculate the mean of the nth attribute offset $m_n = \frac{1}{M}\sum_{i=1}^{M} x_{ni}$. Then, we normalize m_n through dividing it by $\frac{\sigma_n}{\sqrt{M}}$, which gives

$$A_n = \frac{\sqrt{M} m_n}{\sigma_n} \sim N(\frac{\sqrt{M} x_n}{\sigma_n}, 1), \quad (36)$$

New decision variable S is attained through weighted-sum of the N selected attributes.

$$S = \sum_{i=1}^{N} w_i A_i \sim N(M \sum_{i=1}^{N} \frac{w_i x_i}{\sigma_i}, \sum_{i=1}^{N} w_i^2), \quad (37)$$

As a result, we can utilize S to decide the correct hypothesis. In the end, the logarithm of likelihood ratio test (LRT) is used to make decision. After computation [17] we get the threshold T:

$$T = \sqrt{\sum_{i=1}^{N} w_i^2} Q^{-1}(P_{FA}) > 0, \tag{38}$$

Therefore, the detection probability is obtained:

$$P_D = P\{S > T|H_1\} = Q\left(Q^{-1}(P_{FA} - \frac{\sqrt{M}\sum_{i=1}^{N}\frac{w_i x_i}{\sigma_i}}{\sqrt{\sum_{i=1}^{N} w_i^2}})\right), \tag{39}$$

Our goal is to maximize the detection probability (i.e., the cost function $f(w)$ given by 40):

$$f(w) = \frac{\sqrt{M}\sum_{i=1}^{N}\frac{w_i x_i}{\sigma_i}}{\sqrt{\sum_{i=1}^{N} w_i^2}}, \tag{40}$$

After computation [17], the optimal weights are given:

$$w_k = \frac{x_k}{\sigma_k C}. \tag{41}$$

3 Orthogonal Frequency Division Multiplexing

OFDM is a Frequency Division Multiplexing technique that has become an important multi-carrier modulation scheme used in several standards such as IEEE 802.11 WLAN, WiMaX and 3GPP LTE [8].
The multi-carrier approach makes OFDM a robust technology since it can easily deal with Inter-Symbol Interference (ISI), severe channel conditions and provide high-performance methods for channel estimation.
In this section, an overview about OFDM in the IEEE 802.11a standard and the basics of OFDM is provided [9], [10].

3.1 OFDM multi-carrier modulation

OFDM is a mechanism that multiplexes a set of symbols over orthogonal carriers in the frequency domain. First, the total bandwidth is divided in K carriers, each of which is associated to one symbol.
The carriers are exponential functions which are orthogonal (see Fig. 4), a fact that allows the receiver to detect the K symbols from the signal.

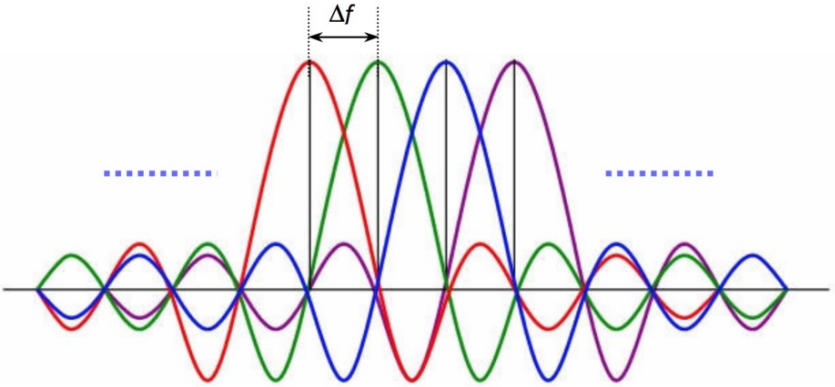

Figure 4: Orthogonal sub-carriers, frequency domain

The operation of multiplexing data with the exponential carriers is equivalent to computing the IFFT of the input sequence, and this operation is performed by the transmitter. Similarly, the FFT is computed in the receiver to obtain the transmitted symbols.
Each component of the FFT is associated with the symbol in the carrier of the same

index, thus K being the length of the FFT and equivalently the total number of carriers in one OFDM symbol. Fig. 5 shows the typical OFDM transmitter in GNU Radio. Note that a cyclic prefix is appended to the reverse FFT (IFFT) in order to avoid ISI.

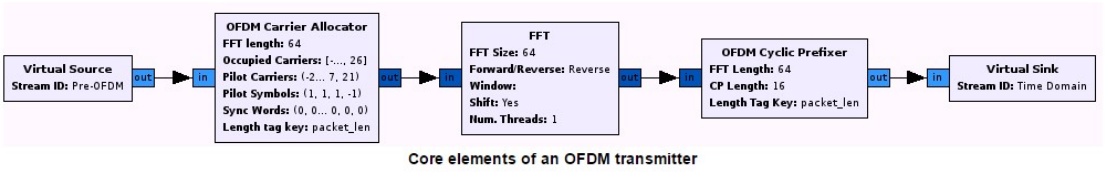

Figure 5: GNU Radio OFDM transmitter

The receiver in Fig. 6 contains blocks related to Channel Estimation and equalization as well, aiming at minimizing the effect of the channel and frequency offset in the received signal.

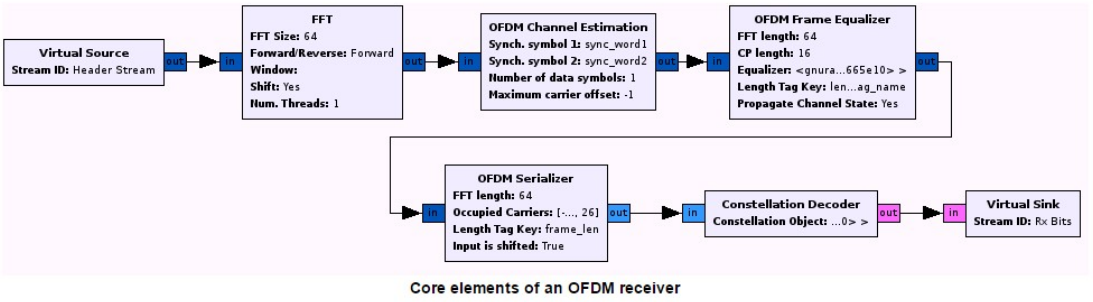

Figure 6: GNU Radio OFDM receiver

There are two ways of estimating the channel in OFDM:

1. Using the Long Training Sequence, the channel estimation in the 52 frequencies is obtained once every frame.
2. Using the pilot sequence, the estimation of the 4 sub-carriers is obtained once every OFDM symbol. This configuration updates the channel estimate faster, but an interpolation method must be applied in order to obtain the estimates in the rest of the frequencies.

3.2 Signal model and channel estimation

The transmitted OFDM symbol in $t = 0$ is modeled as:

$$s_0(n) = \sum_{l=0}^{K-1} X(k) e^{\frac{2\pi l n}{K}} = KIDFT[X(k)]$$

Where $X[k]$ is a sequence of symbols modulated with the chosen sub-carrier modulation.
Then, the received signal in the first symbol interval $0 <= t <= T_u$ is:

$$r_x(t) = s_0(t) * h(t) + n(t)$$

Where $n(t)$ is zero-mean Gaussian noise added by the wireless channel.
However, since we added a cyclic prefix to the transmitted signal, the convolution is modeled as circular and that allows the signal to be the following after computing the FFT:

$$R[k] = X[k]H[k] + N[k]$$

Supposing that the carriers are perfectly orthogonal and that ISI is perfectly eliminated.
Then:

$$\hat{H}[k] = \frac{R[k]}{X[k]}$$

- If $X[k]$ is the long training sequence, we can estimate the channel in the 52 frequencies.
- However, when $X[k]$ is a data OFDM symbol, we still know X[k] in $k = \{-21, -7, +7, +21\}$ (i.e., the pilot positions), and therefore the channel is obtained too. The other frequencies are estimated by performing an interpolation.

Several adaptive estimation algorithms such as Least Squares (LS), Least Mean Squares (LMS) among others are used to obtain better estimates of the channel in the rest of the time instants and sub-carrier indexes.

3.3 IEEE 802.11 burst

In 802.11a, the OFDM symbols are encapsulated in frames or bursts (see Fig. 7 and Fig. 8).
The amount of transmitted data per frame varies over time and can be specified for

each application, depending on the desired data rate and bit error probability.
Each OFDM symbol consists of a total of 64 sub-carriers, containing modulated information. The data sub-carrier modulation is either BPSK, QPSK, 16QAM or 64QAM, and can vary from one burst to another.
The Physical Layer Convergence Protocol (PCLP) Preamble consists of both a short and a long training sequences, which are known by the transmitter and receiver. Those are used for synchronization, frequency offset and channel estimation. The signal field contains information about the modulation scheme.
Each OFDM data symbol contains 4 BPSK pilots, which are used for further offset and channel estimation, just like the training sequences at the beginning of each frame. Those values vary from one OFDM symbol to another following a pseudo-random sequence, specified in the standard.

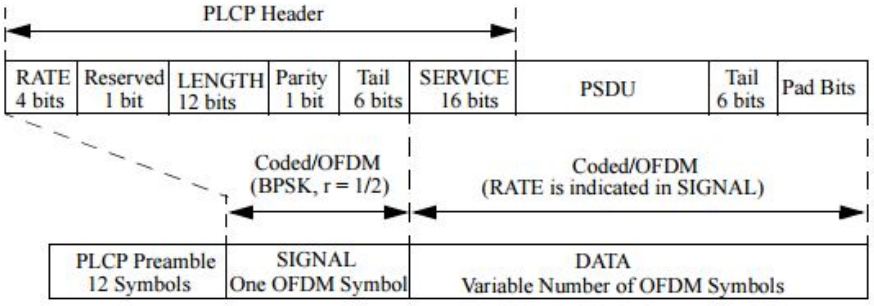

Figure 7: OFDM Burst in IEEE 802.11a, from [8].

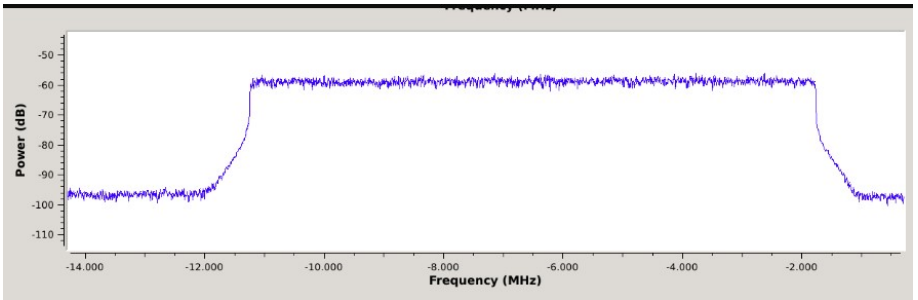

Figure 8: Frequency characterization of a burst

4 USRP and GNU Radio

To test our simulations and verify that physical layer authentication is feasable in real wireless networks, we used Software Defined Radios (SDR) in conjuction with GNU Radio.

4.1 Environment Setup

In GNU Radio we used the module IEEE 80211, also called the Wime Project [11], which contains all the necessary signal processing blocks to both transmit and receive WiFi signals. This was coupled with a setup of three USRPs, where the receiving one was placed in the middle between the two other radios with a couple of meters distance between them. This was done to create an environment where the channels would have similar conditions in respect to the receiver, so an observed difference in channel estimation would be reproducible in most scenarios. The tools involved during our implementation include Ubuntu 16.04, GNU Radio, Python 2.7 and related packages, C++ IDE, and Wireshark 2.8. The installation and debugging sheets regarding our field testing are provided as attachments in the project web page.

Besides, to enable fast prototyping, we also wrote a shell scrip named *init_usrp.sh* which automatically deals with ports opening, promisc mode setting and IP allocation for the selected port.

4.2 Off-line Measurements

The WiFi transmitter and receiver we used is compliant with the IEEE 802.11p protocol [12]. For offline purposes, it suffices to include the existing RFtap [13] package which connects to the network analyzer software in order to extract the parameters such as frequency offset. However, to fully control the data flow, we develop our own custom blocks in both python and C++ in order to format the received message.

The workflow of developing Out Of Tree (OOT) modules in GNU Radio is detailed in the official documentation of GNU Radio [14]. Following is a short summary of the workflow, leaning on terms from the official GNU Radio documentation. First, with GNU Radio module tools our own module is created, specifying the programming language, the QA test and the input.

Then, with python or C++, the main functionality of the block is implemented in the respective files created by the module tools. In the xml file, include the information with the GNU Radio platform and compile & build the block. The sample xml prototype for our buffer block is shown below. In this way, the custom block will be embedded into the GNU Radio graphical interface.

```
<?xml version="1.0"?>
<block>
  <name>EQ2440Buffer</name>
  <key>EQ2440Module_EQ2440Buffer</key>
  <category>[EQ2440Module]</category>
  <import>import EQ2440Module</import>
  <make>EQ2440Module.EQ2440Buffer()</make>
  <sink>
    <name>bufferin</name>
    <type>message</type>
  </sink>
  <source>
    <name>bufferout</name>
    <type>message</type>
  </source>
</block>
```

Frequency offset and SNR are encapsulated in a pmt pair (car and cdr). At the receiver, we implemented a custom block that extracts the data from the car of the pair, which is a dictionary. The source and destination MAC addresses are encapsulated as uint8 bytes in the cdr section of the pmt pair, and then these bytes are extracted according to the WiFi data frame structure and written to a file. As an example, the sample data format during our extraction is shown below.

```
dst mac: ff:ff:ff:ff:ff:ff | src mac:58:bf:ea:77:a9:2e | snr:17.6305 | frequency offsets:-17186.7
dst mac: 98:6c:f5:16:63:ba | src mac:58:bf:ea:77:a9:2f | snr:16.3381 | frequency offsets:-15200.4
dst mac: 58:bf:ea:77:a9:2f | src mac:98:6c:f5:16:63:ba | snr:22.7316 | frequency offsets:-33812.6
dst mac: 98:6c:f5:16:63:ba | src mac:58:bf:ea:77:a9:2f | snr:14.4812 | frequency offsets:-13021.3
```

This file is then post-processed with MATLAB [15].

The channel estimates are extracted from an edited version of the OFDM Frame Equalizer of the Wime Project. For each received OFDM frame, the channel estimation obtained from the training sequence is written to file and analyzed in MATLAB.

4.3 Real-time Implementation

For real-time purposes, extra functionalities with custom blocks are introduced. These functionalities include, tuning settings at Tx to make Eve appear as Alice for the receiver, e.g., faking features such as frequency offsets, MAC address, IQ imbalances, and real-time identity decision for different features at Rx with pre-determined weights, perfection of the interfaces, e.g., real-time plots of features at Rx, animation of USRP identity for demo purposes at Tx, among others. Besides, alarm beeping functionality with a custom buffer at Rx is also introduced.

The Buffer is placed at the very end of the receiver to counteract the fact that even when Alice is transmitting, a few samples will occur outside the threshold and appear as Eve. Thus we make sure that a sequence of a certain length from the decision making function only contains Eve, before we actually let the Eve-decision pass through.

The decision making function is where we combine the different features. This is done by letting each feature decide if it is Alice or Eve independently. Then we sum each output, 1 for Alice and 0 for Eve, multiplied by its weight. This quantity is then divided by the total sum of the weights to obtain a normalized value. If this final value is above 0.5, meaning more than half of the features agree that it is Alice, we decide it is Alice.

The real-time plots at the receiver need to be fast, while also quite simple as it is just for demonstration purposes. The python library PyQtGraph offers high speed and performance, while offering a simple interface for creating plots with the graphical framework Qt. Below is a simplified example of how we create a plot with PyQtGraph.

```
import from pyqtgraph.Qt import QtGui, QtCore
import pyqtgraph as pg

app = QtGui.QApplication([])
win = pg.GraphicsWindow(title="eq2440 plotting")
snr = win.addPlot(title="SNR plot")
snr_curve = snr.plot()

def update():
    snr_curve.setData(data)
```

In order to enable Eve to fake the features when transmitting, here are the basic underlying ideas. For MAC address, it is modified real-time by extending the hierarchical PHY layer block of Tx to set the MAC address. Such MAC addresses can

be changed from the GUI interface and embedded into the data frame in real-time before transmitting the messages. As for frequency offsets, we add an offset of the carrier frequency in the UHD block to be modified in real-time.

The animation for the Transmitter identity is introduced with pygame package [16] for demo purposes. The corresponding figure for Alice or Eve will blink as shown in Fig. 9 screen snap. This is implemented in our custom EQ2240animation block.

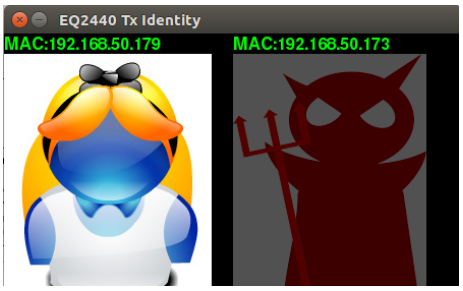

Figure 9: Tx Identity Animation

To achieve the alarm beeping and reduce the false alarm probability, a custom block is developed in C++ with input of the identity decision and output of the frequency tune which is fed to the audio sink, as seen in Fig. 10. A proper buffer size should be set to reduce the false-alarm probability. In GNU Radio, the decision and the frequency values are written as pmt asynchronous messages to be transmitted to the next block.

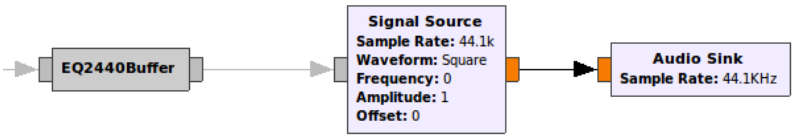

Figure 10: Buffer Block Implementation

5 Results

In this section, both the simulated and experimental results are detailed. The simulations are available in [18], and they are based on two different scenarios: OFDM and simplified QPSK system. Four cases are considered: CFO, IQI, channel estimation and feature combination.

The scenario displayed in Fig. 11 is evaluated using the P_{FA} and P_{MD} for the experimental case. For more details, refer to the video in [19]. Four cases are considered: CFO, SNR, channel estimation and feature combination. The distributions for the default properties of the radios are analyzed. Two transmitters (i.e., Alice and Eve) are located 1 meter from a receiver (i.e., Bob), the SNR of the link is 15dB.
A two-step process has been followed in order to provide a real-time implementation of our prototype.

- First, the data from the receiver is collected and post-processed using MATLAB. The different distributions are evaluated and the hypothesis testing thresholds are decided.
- Finally, once the features for Alice are learned by the receiver and the thresholds are set, the authentication algorithms are implemented in real-time.

Next, the post-processing analysis is detailed for the specific scenario. However, other cases will be considered as well in the final demonstration.

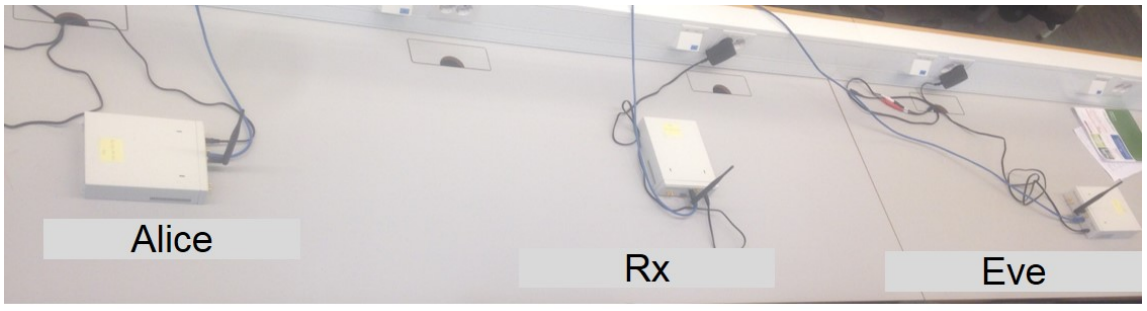

Figure 11: Scenario with three USRP

5.1 Channel Estimation

For the simulations, we implemented a basic channel estimation method based on the pilots for each OFDM symbol, as explained above. The different miss-detection probabilities depending on the required false alarm and the SNR of the communication link are displayed in 12, which stay within the range of 5-20%.

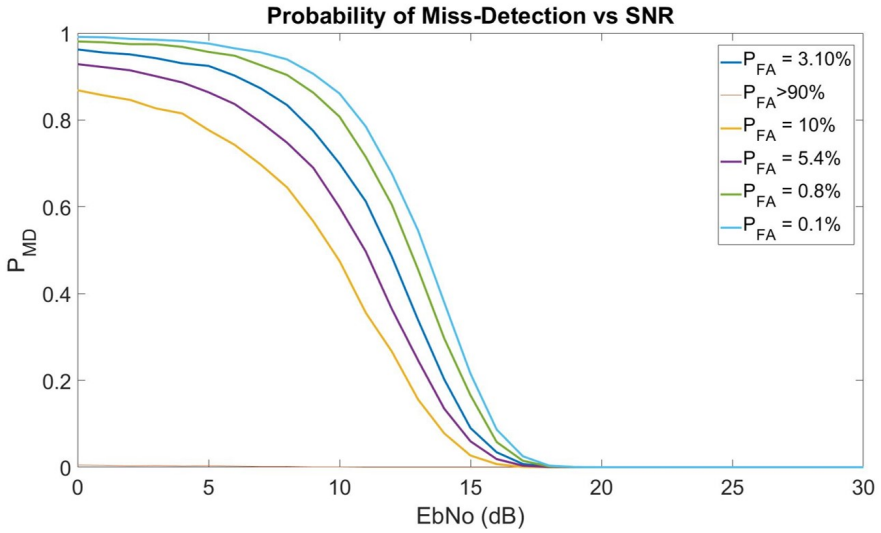

Figure 12: Simulated SNRs vs P_{MD} with different false alarm rate P_{FA}

In GNU Radio, an estimate of the channel frequency response is provided using different methods such as linear combination, LS and LMS. Since the estimation method is more advanced, the results for the practical case are slightly better than our simulations. Also, since the SDR are 1m away from each other, the channels differ more than in our simulations.
As explained in Section 3.2, two different algorithms are provided for channel estimation in the IEEE 802.11a standard. The LMS method has been chosen, which uses the long training sequence for adaptive channel estimation once every OFDM burst.
A similar procedure as the one stated in Section 2.3 has been implemented. For simplicity, σ^2 is not estimated and hence, L does not follow the Chi Square theoretical distribution. However, as shown in Fig. 13, both L_{Alice}, L_{Eve} are distinguished clearly and different thresholds can be set in order to evaluate the performance of the algorithm.

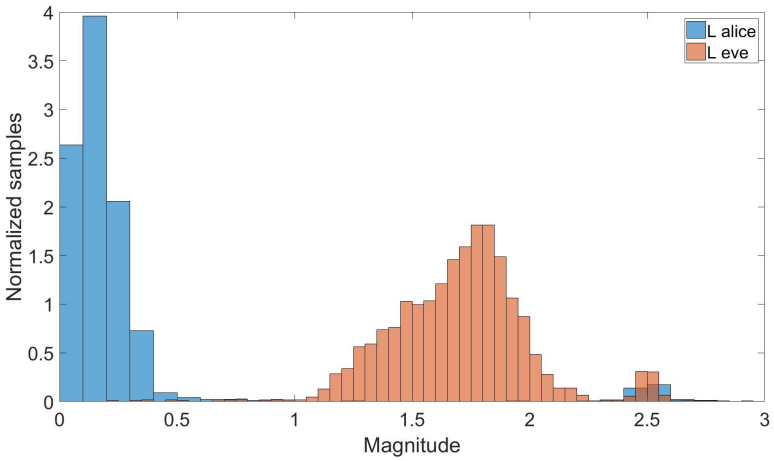

Figure 13: Histograms for channel estimation

Note that, due to some sporadic reception errors represented on the right side of the histograms, the probability of false-alarm, which is displayed as the red line in Fig. 14 is always higher than 4%. In the real-time implementation we use a buffer in the receiver to reduce that probability.
Finally, Fig. 15 demonstrates the probability of miss-detection as a function of the probability of false-alarm.

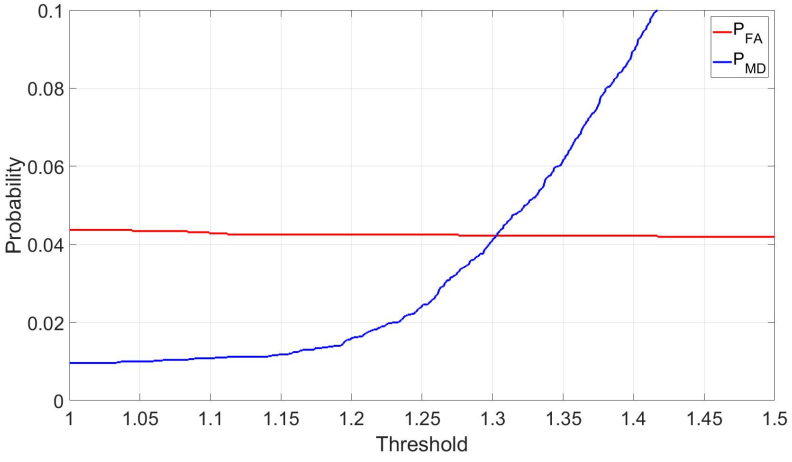

Figure 14: Thresholds for channel estimation

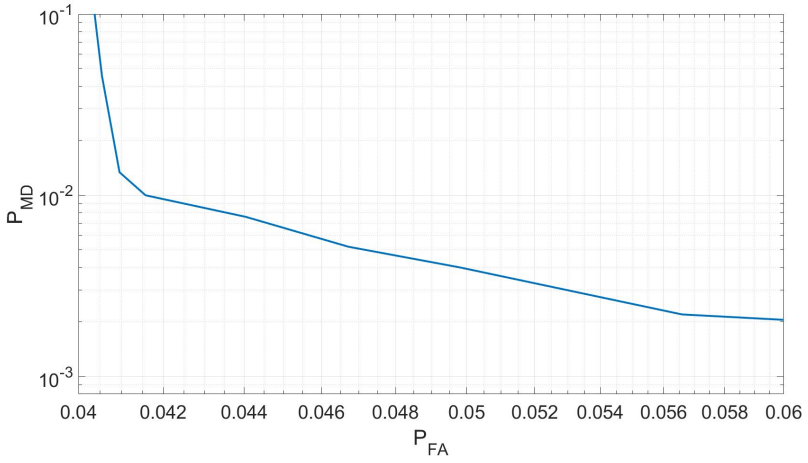

Figure 15: False Alarm and Miss Detection for channel estimation

The channel estimation feature is really difficult to fake, unlike the CFO or SNR, which can be manipulated by the transmitter. However, the transmitters need to be far enough from each other so that the channels can be distinguishable.

5.2 Carrier Frequency Offset (CFO)

The results for the simulations are displayed in Fig. 16. As we can see, the miss-detection probability can take values below 5%.

In a similar manner as the channel case, the CFO estimates are provided in the synchronization block of the GNU Radio flow chart. More specifically, both the short and long training sequences are used to estimate the CFO.
Similarly as in our simulations, the CFO follows a Gaussian distribution due to both the estimation error and the wireless channel (see Fig. 17).

Fig. 18 and Fig. 19 provide the evaluation of P_{FA}, P_{MD} with more detail. The results are slightly better than in the Channel Estimation case, especially regarding the false-alarm probability.

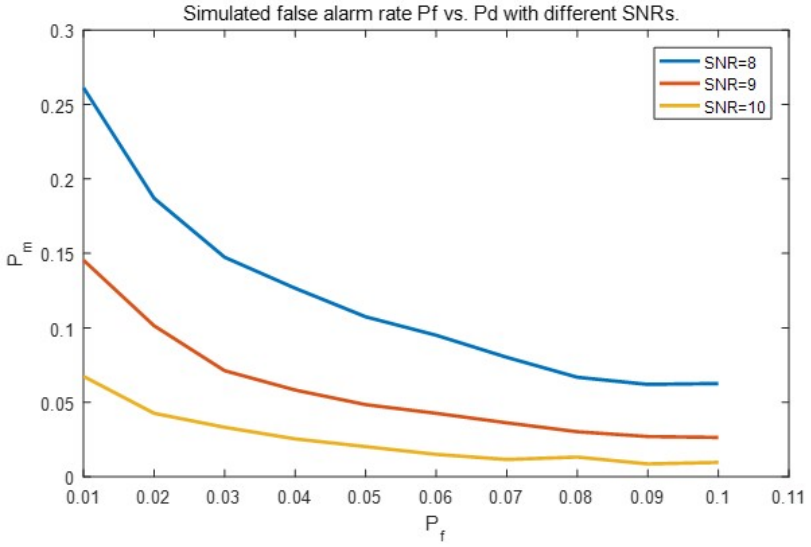

Figure 16: false alarm rate Pf vs. Pd with different SNRs

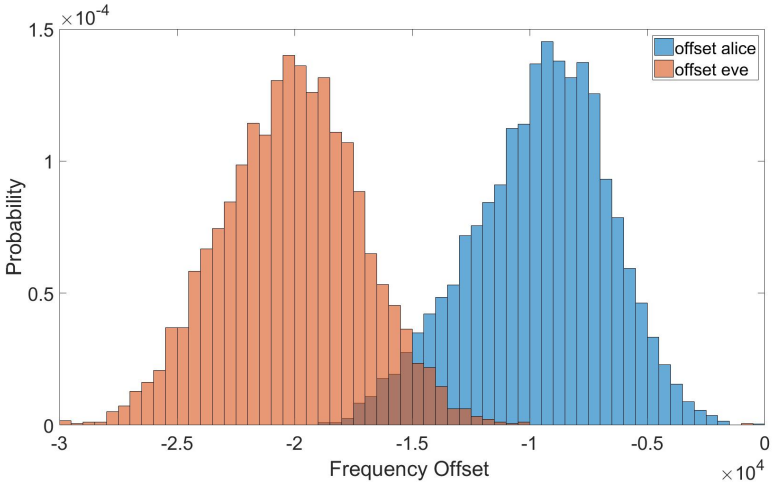

Figure 17: Histograms for CFO

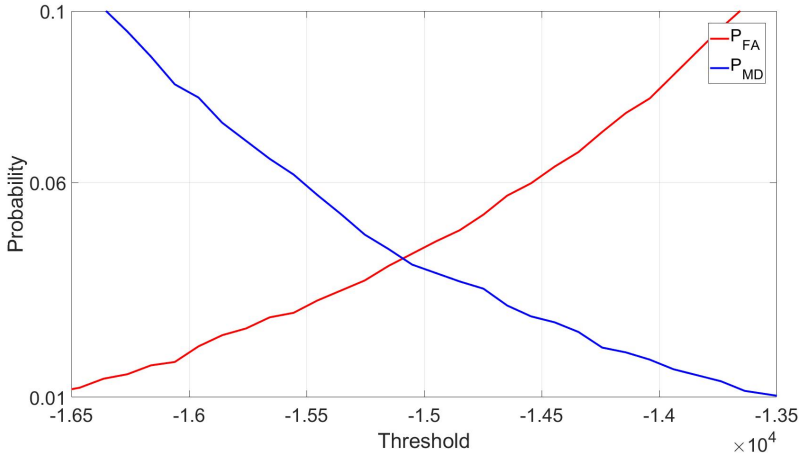

Figure 18: Thresholds for CFO

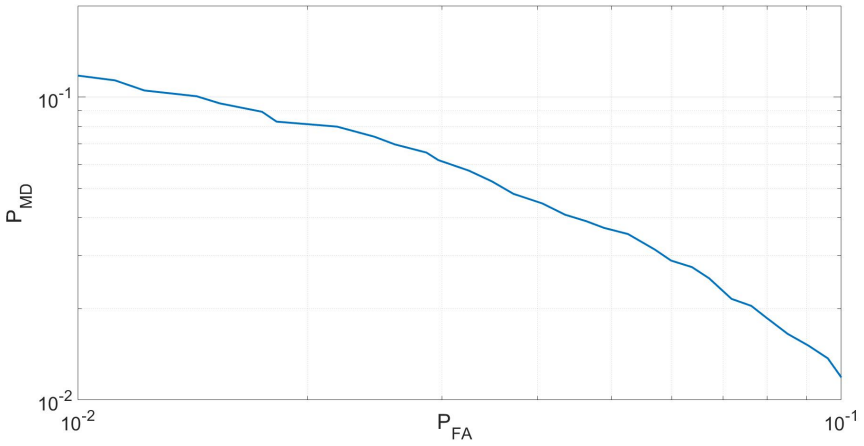

Figure 19: False Alarm and Miss Detection for CFO

The CFO can be easily manipulated by the transmitter. However, in order to be miss-detected, the attacker (Eve) must know the CFO between Alice and Bob, and Bob and him.

5.3 Signal to Noise Ratio (SNR)

Once the transmitted symbols are computed in the receiver using a frame equalizer, the signal and noise powers are calculated, and the SNR is estimated. The default values for both transmitters are shown in Fig. 20.

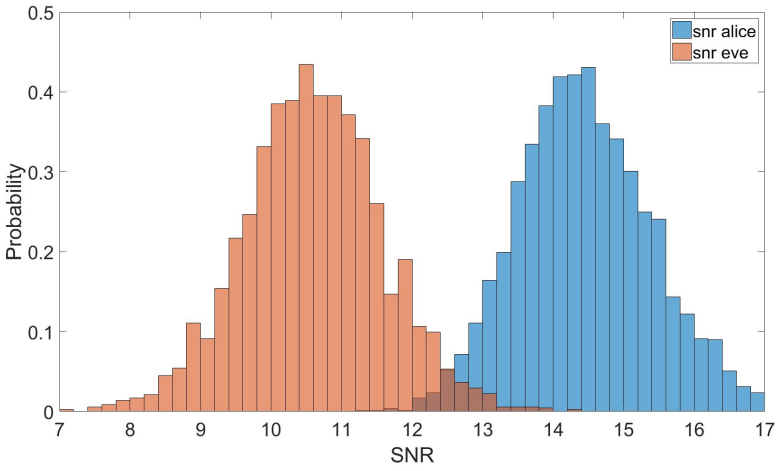

Figure 20: Histograms for SNR

Then, the thresholds are evaluated and the plots for the P_{FA}, P_{MD} are displayed in Fig. 21 and Fig. 22.
As in the CFO case, the SNR can be faked by an attacker. However, the specific information of the communication link is needed.

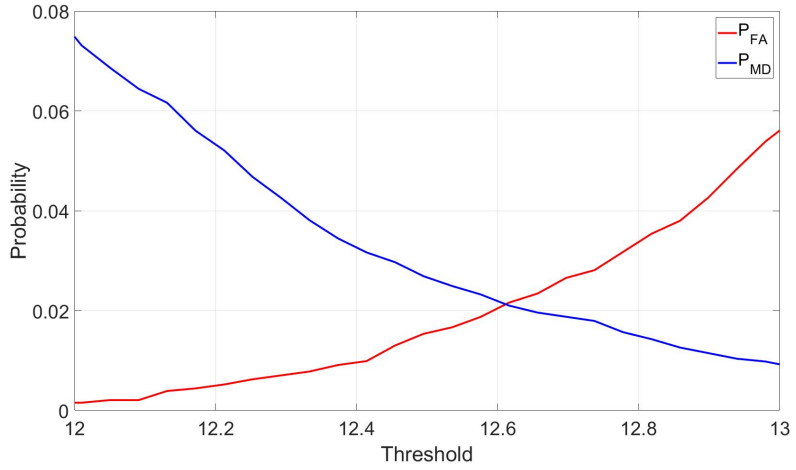

Figure 21: Thresholds for SNR

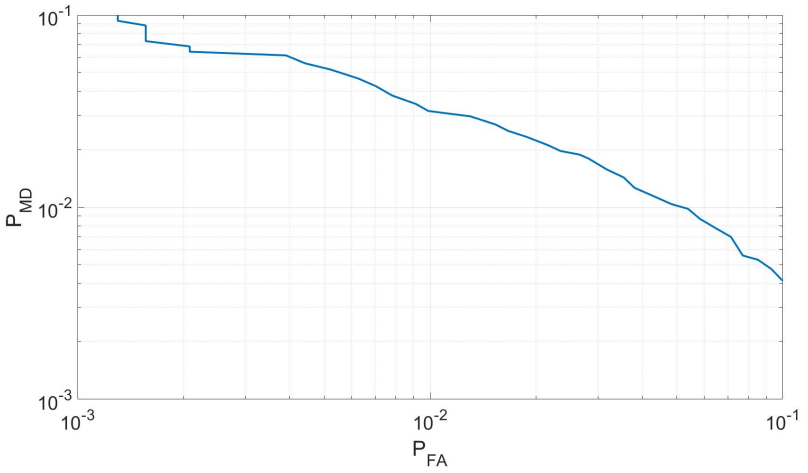

Figure 22: False Alarm and Miss Detection for SNR

5.4 IQ Imbalance

In Fig. 23, the simulated results are demonstrated.

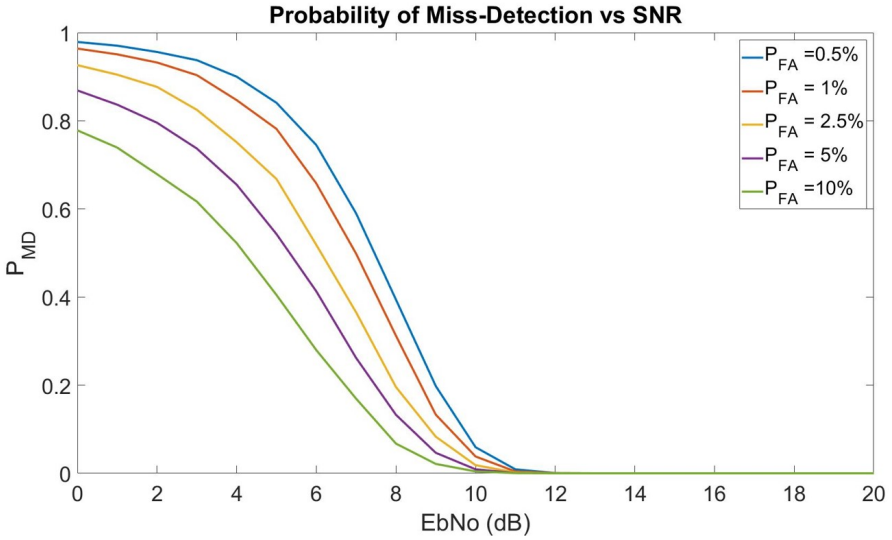

Figure 23: Simulated SNRs vs P_{MD} with different false alarm rate P_{FA}

5.5 Feature Combination

Fig. 24 provides a comparison between the different features evaluated above in the experimental case.
The CFO and SNR follow a similar trend, SNR being slightly better. However, the SNR just applies to the invariant case while the CFO is not so sensitive to mobility. Channel Estimation provides always a probability of false-alarm higher than 4%, but a lower ratio than the other features. That is because 52 different frequencies are taken into consideration, hence improving the accuracy.

The simulation results for the explained feature combination algorithm are displayed in Fig. 25, Fig. 26.

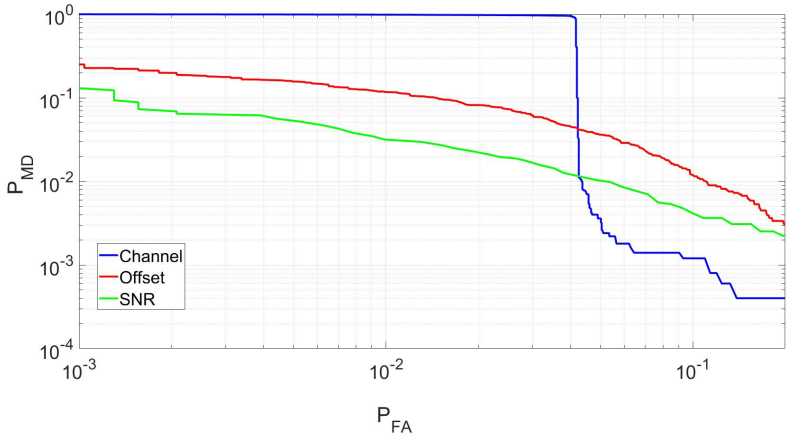

Figure 24: Comparison between features

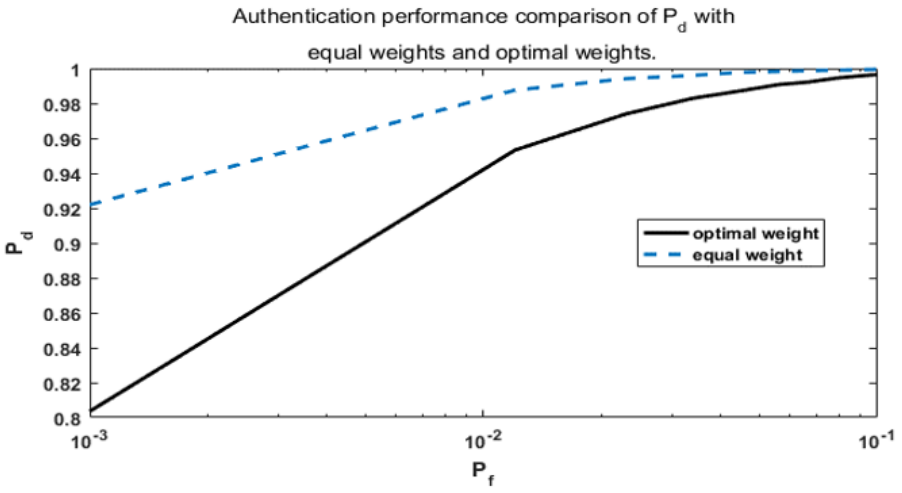

Figure 25: Authentication performance comparison of P_D with equal weights and optimal weights.

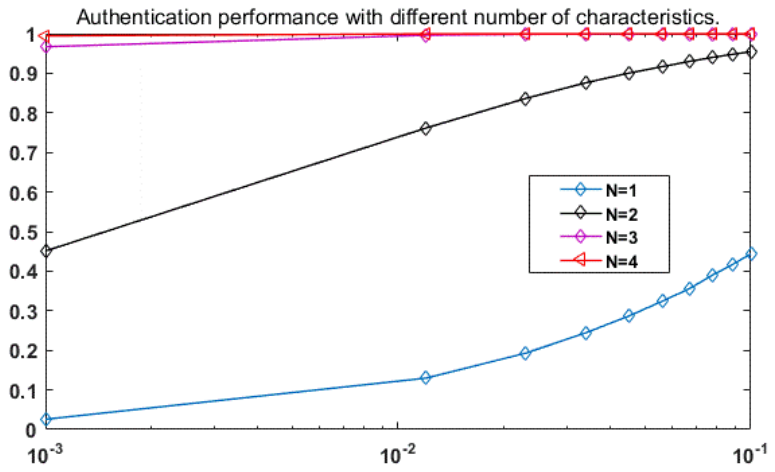

Figure 26: Authentication performance comparison of P_D vs P_F with different numbers (N) of characteristic

For the real-time implementation explained above, all combinations of weights are plotted in Fig. 27 using the optimal thresholds for each feature.
For instance, setting the false-alarm probability to 2% leads to a possible span of miss-detection between 2.4% and 5.6%, each point corresponding to a different combination of weights. In this case, the proper weights would be set to minimize the miss-detection to 2.5%. In other words, by previously setting the optimal thresholds, all the combinations displayed in the plot can be achieved by setting different weights.

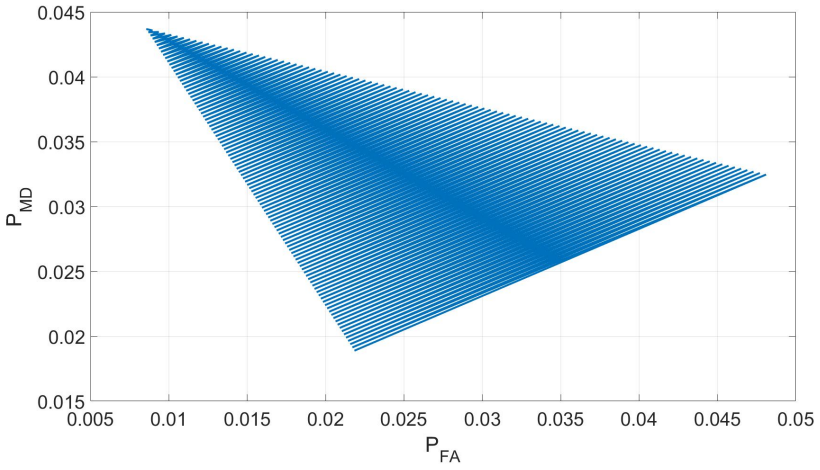

Figure 27: Optimization using weights

Then, by providing a cost function, the weights can be optimized. This function would depend on the required robustness needed (i.e., the required miss-detection probability and the false-alarm that can be tolerated).

6 Conclusions

This report focuses on Physical Layer Authentication techniques to enhance existing key-based cryptography algorithms implemented in higher layers of the network.

Analog Front-End (AFE) imperfections, such as In-Phase Quadrature Imbalance (IQI), Carrier Frequency Offset (CFO), Signal to Noise Ratio (SNR) and Received Signal Strength (RSS), exploit hardware characteristics of the transmitter receiver pair. The probabilities of miss-detection and false-alarm are usually below 5%. However, if a user obtains certain information from the legitimate transmitter and receiver, these features can easily be faked.

Channel-based features (just the invariant case was considered), exploit the fading characteristics of a wireless channel. If the transmitters are far enough, the frequency response of the channel is very difficult to fake. However, more computational power is needed to provide adaptive estimation techniques, which are usually very sensitive to mobility. The miss-detection probability goes down to 0.1%. However, due unresolved system limitations, we cannot achieve a false-alarm probability lower than 4%.

Since all features have their limitations, an optimal combination is usually case-dependent. For example, channel-based features do not perform well if an attacker is close to the legitimate user. For that reason, a combination algorithm based on weights has been analyzed in this project, and a simplified version is implemented in real-time using USRP and the GNU Radio Software.

The authentication procedure follows: firstly, the data from off-line measurements is obtained in order to set the required authentication parameters (i.e., thresholds for each feature and optimal weights). Then, the authentication algorithm is executed in real-time in the static or invariant scenario.

Further research can be conducted regarding real-time learning techniques to continuously update the features of the legitimate user, evaluate the performance of the algorithms and select an optimal weight combination. Also, variant channel estimation techniques might be considered [20] to improve the performance of the channel-based features.

References

[1] G. J. Simmons, "A survey of information authentication," Proc. IEEE, vol. 76, no. 5, pp. 603–620, May 1988.

[2] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, Handbook of Applied Cryptography. Boca Raton, FL: CRC, 2001

[3] X. Wang, P. Hao and L. Hanzo, "Physical-layer authentication for wireless security enhancement: current challenges and future developments," in IEEE Communications Magazine, vol. 54, no. 6, pp. 152-158, June 2016. doi: 10.1109/MCOM.2016.7498103

[4] Mouli C., Singh S. (2011) Physical Layer Authentication in Wired and Wireless Communication Systems. In: Nagamalai D., Renault E., Dhanuskodi M. (eds) Advances in Digital Image Processing and Information Technology. Communications in Computer and Information Science, vol 205. Springer, Berlin, Heidelberg

[5] L. Xiao, L. Greenstein, N. Mandayam and W. Trappe, "Fingerprints in the Ether: Using the Physical Layer for Wireless Authentication," 2007 IEEE International Conference on Communications, Glasgow, 2007, pp. 4646-4651. doi: 10.1109/ICC.2007.767

[6] P. Hao, X. Wang and A. Behnad, "Relay authentication by exploiting I/Q imbalance in amplify-and-forward system," 2014 IEEE Global Communications Conference, Austin, TX, 2014, pp. 613-618. doi: 10.1109/GLOCOM.2014.7036875

[7] W. Hou, X. Wang and J. Y. Chouinard, "Physical layer authentication in OFDM systems based on hypothesis testing of CFO estimates," 2012 IEEE International Conference on Communications (ICC), Ottawa, ON, 2012, pp. 3559-3563. doi: 10.1109/ICC.2012.6364429

[8] "IEEE 802.11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications" (PDF). (2012 revision). IEEE-SA. 5 April 2012. doi:10.1109/IEEESTD.2012.6178212.

[9] OFDM in IEEE 802.11 - Overview from Keysight `http://rfmw.em.keysight.com/wireles/helpiles/89600b/webhelp/subsystem/wlan-ofdm./contentm/ofdm_80211-overview.htm`

[10] Basics of OFDM - Overview from Keysight `http://rfmw.em.keysight.com/wireless/helpfiles/89600b/webhelp/subsystems/wlan-ofdm/content/ofdm_basicprinciplesoverview.htm`

[11] `https://www.wime-project.net/`

[12] Fuxjäger, P., et al. "IEEE 802.11 p transmission using GNURadio." 6th Karlsruhe Workshop on Software Radios (WSR). Karlsruhe, 2010.

[13] `https://rftap.github.io/blog/2016/09/01/rftap-wifi.html`

[14] `https://wiki.gnuradio.org/index.php/Guided_Tutoria_GNU_Radio_in_Python`

[15] `https://adamgannon.com/2014/11/18/gnuradio_offline_pt1/`

[16] `http://eyehere.net/2011/python-pygame-novice-professional-1/`

[17] Peng Hao and Xianbin Wang,"Performance enhanced wireless device authentication using multiple weighted device-specific characteristics", Proc.IEEE China Summit and Intl. Conf. Signal and Info. Processing, July 2015, pp. 438–42

[18] `https://www.kth.se/social/group/physical-layer-authe/`

[19] `https://www.youtube.com/watch?v=1EPaTLZO-1w&feature=youtu.be`

[20] L. Xiao, L. J. Greenstein, N. B. Mandayam and W. Trappe, "Using the physical layer for wireless authentication in time-variant channels," in IEEE Transactions on Wireless Communications, vol. 7, no. 7, pp. 2571-2579, July 2008. doi: 10.1109/TWC.2008.070194