Platform for Digital Voice Communication with Channel-Based Key Generation

Jelle Jocqué*, Patrick Van Torre*, Jo Verhaevert*, Hendrik Rogier*
*Ghent University - imec, IDLab, Department of Information Technology (INTEC)
Technologiepark-Zwijnaarde 126, 9052 Gent, Belgium, jelle.jocque@ugent.be

***Abstract*—This paper proposes a novel method for the integration of channel-based key generation in a real-time digital voice communication system. For this purpose, a specific hardware platform was developed. To obtain the keys for the cipher, channel-based key generation is used to generate identical keys on both sides of the channel, based on the unique reciprocal channel characteristics. Hamming code key reconciliation drastically reduces the probability of remaining key errors, while a cyclic redundancy check (CRC) is used as a final checking mechanism. As a proof of concept, the generated key is used to encrypt as well as decrypt the audio by means of an XOR operation at both sides of the link. Several measurements were performed, showing that new keys are continuously being formed and replaced. Additional measurements also create more insight into the key update rate and key entropy for different parameter settings in the algorithm.**

***Index Terms*—Digital Voice Communication, IEEE802.15.4, Channel-Based Key Generation, ADF7242 Transceiver, 2.4 GHz ISM Band.**

I. Introduction

Internet Protocol version 6 (IPv6), over Low Power Wireless Personal Area Networks (6LoWPAN), ZigBee, Bluetooth and Bluetooth Low Energy (BLE) are popular short range network protocols operating in the license-exempt 2.4 GHz ISM radio band. 6LoWPAN is the most commonly used standard in IoT communication protocols, because it is IP-based [1]. IoT applications consist of highly constrained devices that are not always suitable for optimal cryptographic algorithms [2]. Therefore, a more lightweight key algorithm such as channel-based key generation can save processor power and increase battery life of IoT devices.

Channel-based key generation is based on the fact that every channel between two random parties can be seen as a unique communication channel. Because of shadowing, multipath fading and Doppler spread, the two random parties have a unique source of information to build an identical channel-based key. A potential third party, also known as an eavesdropper, does not have access to this kind of information and therefore cannot build the same key.

In literature, a practical wearable communication system implementing channel-based key generation was developed in [3]. The system was tested in different indoor and outdoor scenarios with very promising results. However, it was stated that some kind of key reconciliation is needed to obtain error free keys in all scenarios. More experimental results for an encrypted wireless link between two body-worn sensor nodes are presented in [4]. In [5], an increased accuracy for reciprocal channel assessment was obtained by using a very short round-trip delay of 614 μs. Previous results show the possibility of a very lightweight channel-based key generation algorithm. Although previous papers prove that equal keys can be generated, the implementation of such a key generation algorithm in a real-time digital voice communication system was not documented yet.

In this paper, specific hardware is designed and realised for implementing digital voice communication at 2.45 GHz. Next, a channel-based key generation algorithm is implemented in software to encrypt the audio samples. The encryption algorithm is developed in such way that the voice communication is never interrupted. The software for the microcontroller unit (MCU) is developed in C code.

The remainder of this paper is organized as follows. Section II describes the development of the hardware platform. In Section III the general algorithm for channel-based key generation as well as the implementation in software is explained. Finally, a number of measurements are discussed in Section IV.

II. Hardware Platform

A more detailed overview of the hardware platform is shown in Fig. 1. This hardware platform was developed to function as a standalone system for half-duplex audio communication and it can be divided into four main components. The first one (yellow) consists of two parts, the MCU itself and the serial wire debug (SWD) interface to program it. The RF part consists of the transceiver and some peripherals, and is shown in red. Everything that deals with audio playback or audio recording is shown in blue. Finally, the green part implements the user interface, consisting of an OLED and a number of push-buttons.

A. Transceiver

The ADF7242 transceiver is a highly integrated, low power and high performance transceiver for operation in the global 2.4 GHz ISM band [6]. The transceiver operates in IEEE802.15.4 mode with a maximal data rate of 250 kbps and features two radio frequency input and output (RFIO) channels from which only one will be used.

The balanced output signal of the half-duplex RFIO channel is converted to an unbalanced signal through the 2450BM15A0015E balun from Johanson Technology, with a frequency range from 2400 MHz to 2500 MHz [7]. Next, the unbalanced output port of the balun is connected to the

LFL182G50TC1B905 band-pass filter from Murata Manufacturing and has an impedance of 50 Ω [8]. Finally, the signal goes to a SMA connector launcher, which has also an impedance of 50 Ω.

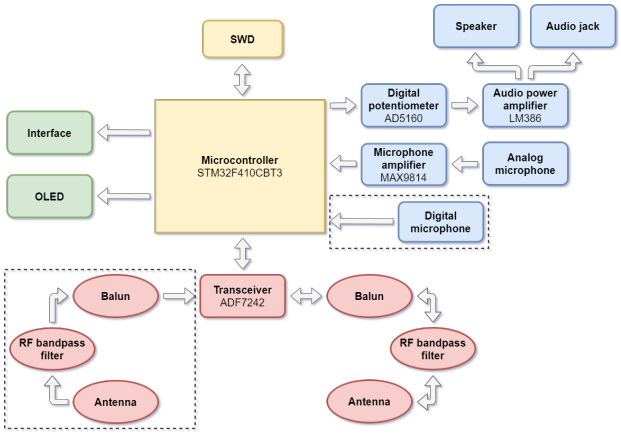

Fig. 1. Detailed block diagram of the standalone hardware platform

B. Audio recording and playback

Audio recording is done with a basic electret microphone which produces an analogue output signal. This signal is then amplified by the MAX9814 microphone amplifier from Maxim Integrated Products [9] and sent to one of the ADC's of the MCU.

The audio can be played on the compact internal speaker or on an external speaker through a 3.5 mm audio jack. Therefore, the DAC of the MCU is connected to an LM386 audio amplifier from Texas Instruments [10]. A digital potentiometer, the AD5160 by Analog Devices [11], is implemented between the DAC output of the MCU and the audio amplifier to be able to control the audio volume in software.

C. User interface

The user interface consists of an OLED display driven by an SSD1306 driver [12] and 5 push-buttons. One of the push-buttons is also connected to the wake-up pin of the MCU. As a result, this button can wake up the MCU from standby mode. Another prerequisite is that one of the push-buttons acts as a push-to-talk button. The remaining buttons can be used for different purposes, e.g. volume control or changing parameters during testing.

D. MCU

The final part of the hardware platform consists of the STM32F410CBT3 MCU [13] and the SWD interface.

III. Channel-based key generation

Channel-based key generation uses the characteristics of the communication channel on which both the transmitter and the receiver are operating. The characteristics of the communication channel, such as fading, are highly correlated between the two legitimate parties. As a result, both parties can use this unique source of information to generate an equal key. A potential eavesdropper will not have access to the same source of information and therefore will not be able to generate the same key.

The first part of this section will give a high-level overview of the software. Next, the general operation of the channel-based key generation algorithm is explained. The last part of this section explains how this algorithm can be implemented in the software for the MCU.

A. Software overview

A high-level overview of the software is shown in Fig. 2. The transmitter uses an internal interrupt to convert the audio samples with the ADC and saves them in a buffer. Next, the audio samples are encrypted with the channel-based key and sent to the transceiver. The transmitter also receives response packages from the transceiver that are necessary for the encryption algorithm.

The receiver reads the encrypted audio packet and saves the decrypted audio samples in a buffer. Similarly, an internal interrupt is used to send the audio samples to the speaker through the DAC.

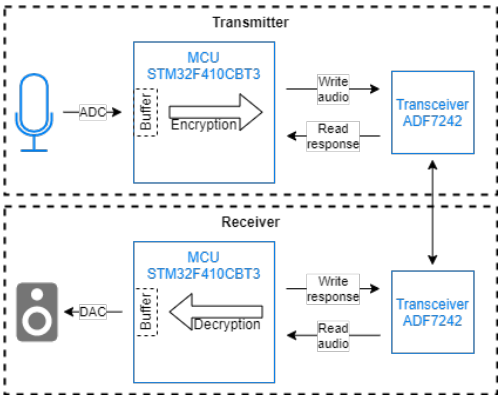

Fig. 2. Software overview

B. General algorithm

The ADF7242 automatically measures and saves the Received Signal Strength (RSS) in one of its registers for every received packet. By using these measurements, a real-time moving average RSS can be obtained for both the transmitter and the receiver. To avoid computational overhead on the processor, an exponential moving average is used to update the mean RSS when a new packet is received. Equation (1) shows the general formula of this algorithm. μ[n-1] corresponds to the previous moving average and μ[n] to the new moving average. RSS[n] is the new incoming value that is added to the exponential moving average. α is a factor between 0 and 1, corresponding to the weight of the last added value. α closer to 1 will lead to a fast changing moving average and α closer to 0 will lead to a slower and steadier moving average.

$$\mu[n] = \alpha\mu[n] + (1-\alpha)\mu[n\text{-}1] \quad (1)$$

New key bits are always chosen by the transmitter. Whenever the RSS measured by the transceiver drops below a certain threshold below the average RSS, a logic 0 will be chosen as a new key bit by the transmitter. Similarly a logic 1 will be chosen when the new RSS is larger than a certain threshold above the average RSS. This decision algorithm is the same as in [4]. When a new key bit was chosen by the transmitter, the receiver needs to be notified as soon as possible. Then the receiver can also determine if the latest RSS value was higher or lower than a certain threshold above or below the average RSS.

In order to build an equal key on both sides of the communication link, two checking mechanisms should be implemented to ensure identical keys are obtained. These checking mechanisms should not send too much sensitive data over the communication channel for obvious security reasons. A first checking and correcting mechanism is the key reconciliation on small groups of bits, as was also proposed in [4]. With Hamming correction this can easily be implemented without too much overhead on the processor. The Hamming code is always calculated by the transmitter and then sent to the receiver. The receiver also calculates the Hamming code and compares this to the one of the transmitter.

Because the mentioned key reconciliation cannot guarantee that identical keys are obtained, a second checking mechanism is required. When a new key was built by the transmitter with a length of e.g. 128 bits, a cyclic redundancy check (CRC) is used as a final check before applying the new key. When the CRC of the receiver is identical to the transmitter's, the new key can be applied. Note that the actual key contains a multiple of the amount of information which is transmitted in the form of Hamming and CRC check bits.

C. Implementation in software

The software for this application was developed in C with the STM32CubeIDE environment of STMicroelectronics. It is mainly interrupt driven to use the MCU as efficiently as possible.

As already mentioned in Section I, a short round-trip delay can increase accuracy for reciprocal channel assessment [5]. Because of that, the transceiver at the transmitter's end is configured in automatic TX-to-RX turnaround mode and similarly the transceiver at the receiver's end is configured in automatic RX-to-TX turnaround mode.

Key reconciliation with Hamming code is applied to every new byte, resulting in a Hamming code of 4 bits. These 4 bits are sent to the receiver to correct one potential wrong bit.

The STM32F410CBT3 MCU features a CRC calculation unit that can be used to calculate the CRC checksum of the key in a very efficient way. Whenever a new key is found the transmitter calculates the CRC32 checksum and adds these 4 bytes to the data packet.

The 128-bit key is currently used to XOR the audio samples, as a proof of concept for voice encryption. This paper focuses on generating equal keys during real-time audio communication without interrupting the audio. In a more critical application, the generated 128-bit key can be used in combination with Advanced Encryption Standard (AES)-128 to achieve a highly secure communication system.

IV. Measurements

All measurements were executed with external monopole antennas with a maximal gain of 4.3 dBi.

A. Spectrum and output power

The behaviour of the ADF7242 transceiver has been verified with a FSV40 spectrum analyser (SA) from Rohde & Schwarz. The measurement of the output power in Fig. 3 was done with the Max Hold functionality of this SA. For this measurement, the peak level of the power amplifier (PA) was programmed to its maximum value of +4,8 dBm. The measured output power at the centre frequency was -14.47 dBm for a resolution bandwidth of 30 kHz. This value is within the expected range for IEEE802.15.4 mode, which has a very high peak-to-average power ratio [6].

Fig. 3. Output power of the ADF7242 transceiver

B. Received signal strength

A number of RSS measurements were done with two devices in a line-of-sight (LOS) and a non-line-of-sight (NLOS) environment. The goal of these measurements was to determine the maximal range for the communication channel in both environments. An overview of the results can be found in Table I.

TABLE I
RSS for different situations

Distance [m]	Propagation	RSS [dBm]
0.1	LOS	-22
1	LOS	-42
5	LOS	-58
10	LOS	-61
20	LOS	-78
5	NLOS	-77
10	NLOS	-89

It can be noticed that as soon as there are large objects in between the two devices (such as walls and doors), the RSS

drops really fast. The sensitivity of the ADF7242 in IEEE 802.15.4 mode is -95 dBm [6]. As can be seen in Table I, a distance of 10 m in a NLOS situation corresponds to a RSS of -89 dBm. This means that the distance between the devices should not exceed this distance in a NLOS situation. In LOS it was possible to reach distances of 20 m without communication loss. Adding a +20 dB gain power amplifier IC, is expected to increase the latter range to about 100 m.

C. Channel correlation between transmitter and receiver

The correlation between the transmitter and the receiver was measured over a period of 8 s in an indoor LOS environment. During the measurements the transmitter was moving the entire time. The RSS samples from both the transmitter and receiver were saved in the flash memory of the MCU and analysed in postprocessing. The packet transmission rate during this test was 400 packets/sec and corresponds to one packet every 2.5 ms. The correlation between transmitter and receiver is shown in a scatter plot in Fig. 4. From the results in Fig. 4, it can be stated that the correlation remains high over the entire range of RSS values. However, there are also outliers in this measurement, which appear more prominent at lower RSS values. The measured RSS for the lowest values is likely varying more due to noise.

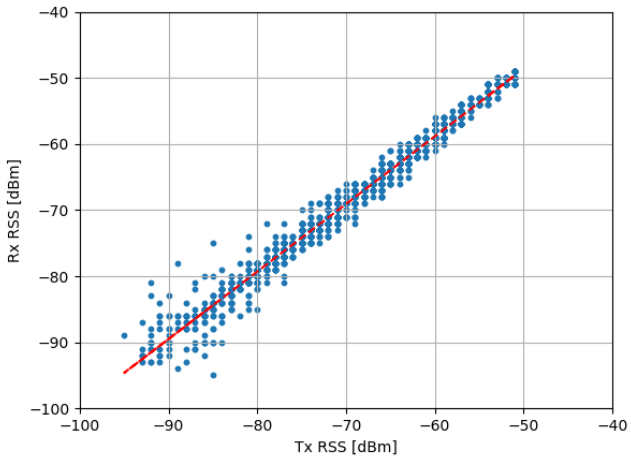

Fig. 4. Correlation between the RSS of the transmitter and the RSS of the receiver in LOS

Similar measurements were performed in both LOS and NLOS environments in [3]. In this paper, the correlation between one of the two legitimate parties and a potential eavesdropper was also examined in different scenarios. Results showed that the correlation between one of the two legitimate users and a potential eavesdropper was significantly lower than between the two legitimate parties.

D. Moving average RSS vs measured RSS

In Fig. 5 the same RSS samples for the transmitter as in the previous measurements are compared against the moving average calculated by the transmitter. The blue line shows the measured RSS samples and the red line shows the average RSS that was updated for every new sample, using α = 0.1. This moving average was calculated with the algorithm explained in Section III. Based on the results in Fig. 5, some conclusions can be drawn. The first remark is that the threshold chosen by the transmitter can have a big impact on the entropy of the key and the update rate of the key. It can be noticed that the dips of the measured RSS are deeper than the peaks upwards. As a result, a big threshold can lead to a key with a lot more logic zeros than logic ones and hence a very low entropy.

The second remark is that the time between choosing two key bits can also have a big impact on the entropy of the key. The packet rate in this measurement was set to 400 packets/sec, which leads to theoretical maximum key update rate of 400 bits/sec. It can be noticed that if no delay is added between choosing two key bits, repetitive patterns of logic zeros or logic ones will lead to a key with a very low entropy.

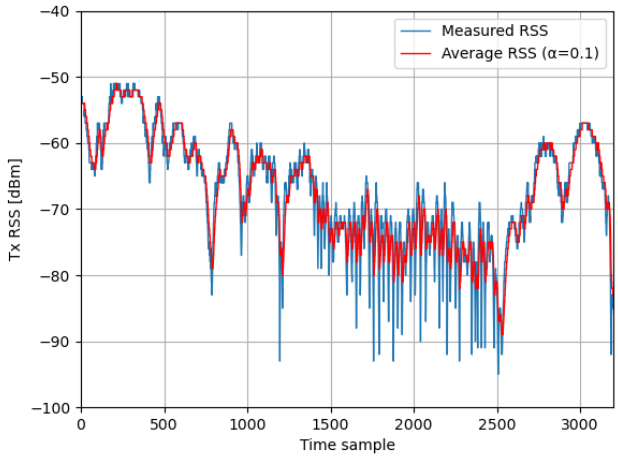

Fig. 5. Measured RSS and average RSS (with α = 0.1) over a time interval of 8 s corresponding to 3200 RSS samples in LOS

E. Key update rate

The speed at which the key is replaced, is an important aspect of this encryption algorithm. The faster the key can be updated, the securer the communication channel becomes.

Two parameters have a large influence on the key update rate. The first one is the threshold, chosen by the transmitter. From the results in Fig. 6, it can be stated that larger thresholds result in lower key update rates. Secondly, the delay between choosing two key bits also has an influence on the key update rate. It is obvious that the longer the transmitter waits after a key bit is chosen, the longer it will take to generate a new key.

Based on the results in Fig. 6, one could choose for no delay and a threshold of 1 dBm to achieve the highest update speed of around 75 bits/sec. But in the next subsection it will become clear that a high update rate also has a major drawback.

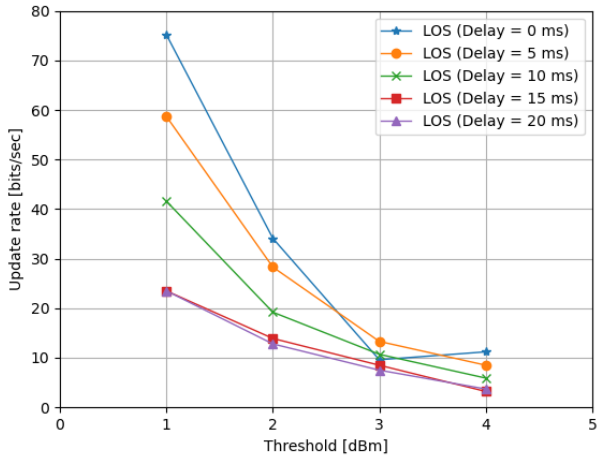

Fig. 6. Key update rate for different thresholds and delays

F. Key entropy

From the results in Fig. 7, it seems that if no delay is used between choosing two key bits, the key entropy per bit is not more than 0.62 bit. This is because the key contains many repetitive patterns of logic zeros or logic ones. These patterns can be avoided by adding a delay after the transmitter has chosen a new key bit.

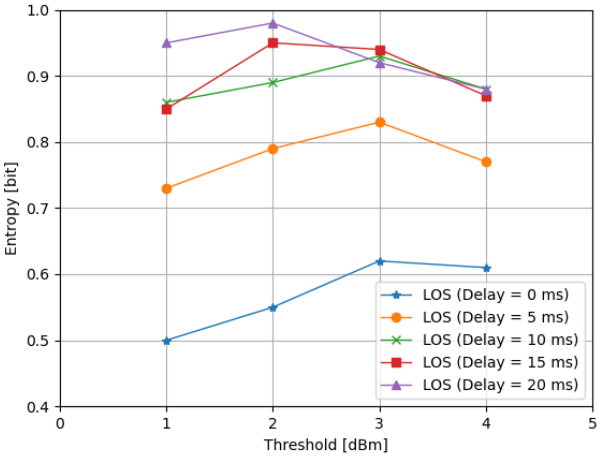

Fig. 7. Key entropy for different thresholds and delays

High key update speeds have the downside of a very low key entropy. Therefore, a minimal delay of 10 ms is necessary to generate good keys. The optimal combination to have a decent update speed and a high entropy is a delay of 20 ms with a threshold of 1 dBm. This combination results in an update speed of around 25 bits/sec and a key entropy of 0.95 bit.

V. Conclusion

In this paper the development of a hardware platform for digital voice communication with channel-based encryption is explained and tested. The goal of this hardware platform was to implement channel-based encryption in real-time digital voice communication on 2.45 GHz. The encryption algorithm was successfully implemented in software without interrupting voice communication at any time. Audio quality was excellent in all tested scenarios.

Different combinations between the key threshold and the delay were tested to achieve an optimal result for both the key update rate and key entropy. An optimal combination of a key threshold of 1 dBm and a delay of 20 ms resulted in an update speed of 25 bits/sec and a key entropy of 0.95 bit.

Based on the results within this paper, the channel-based key generation algorithm can be used in combination with AES-128 to achieve a highly secure digital voice communication channel.

Acknowledgement

This work was partly funded by the Research Foundation-Flanders (FWO) through the "MUlti-SErvice Wireless NETwork", FWO-FRS Excellence of Science – EOS project.

References

[1] S. Al-Sarawi, M. Anbar, K. Alieyan, and M. Alzubaidi, "Internet of Things (IoT) Communication Protocols: Review," in *2017 8th International Conference on Information Technology (ICIT)*, 2017, pp. 685–690.

[2] R. Roman, P. Najera, and J. Lopez, "Securing the Internet of Things," *Computer*, vol. 44, no. 9, pp. 51–58, 2011.

[3] P. Van Torre, "Channel-Based Key Generation for Encrypted Body-Worn Wireless Sensor Networks," *SENSORS*, vol. 16, no. 9, 2016. [Online]. Available: http://dx.doi.org/10.3390/s16091453

[4] P. Van Torre, T. Castel, and H. Rogier, "Encrypted Body-to-Body Wireless Sensor Node Employing Channel-State-Based Key Generation," in *2016 10th European Conference on Antennas and Propagation (EuCAP)*. IEEE, 2016, pp. 1–5. [Online]. Available: http://dx.doi.org/10.1109/EuCAP.2016.7481769

[5] P. Van Torre, Q. Van den Brande, J. Verhaevert, J. Vanfleteren, and H. Rogier, "Key Generation Based on Fast Reciprocal Channel Estimation for Body-Worn Sensor Nodes," in *Proceedings of the European Conference on Antennas and Propagation*, 2017, pp. 293–297.

[6] Analog Devices, "Low Power IEEE 802.15.4/Proprietary GFSK/FSK Zero-IF 2.4 GHz Transceiver IC," 2010. [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/ADF7242.pdf

[7] Johanson Technology, "High Frequency Ceramic Solutions," 2013. [Online]. Available: http://www.farnell.com/datasheets/1722470.pdf?_ga=2.217718422.1017368256.1585411675-1958453732.1576777744

[8] Murata, "LFL182G50TC1B905," 2018. [Online]. Available: https://www.mouser.be/datasheet/2/281/murata_mure-s-a0006330872-1-1740366.pdf

[9] Maxim Integrated Products, "Microphone Amplifier with AGC and Low-Noise Microphone Bias," 2016. [Online]. Available: https://datasheets.maximintegrated.com/en/ds/MAX9814.pdf

[10] Texas Instruments, "LM386 Low Voltage Audio Power Amplifier," 2017. [Online]. Available: http://www.ti.com/lit/ds/symlink/lm386.pdf

[11] Analog Devices, "256-Position SPI-Compatible Digital Potentiometer AD5160," 2014. [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/AD5160.pdf

[12] Solomon Systech Limited, "SSD1306," 2008. [Online]. Available: https://cdn-shop.adafruit.com/datasheets/SSD1306.pdf

[13] STMicroelectronics, "STM32F410x8 STM32F410xB," 2013. [Online]. Available: https://www.st.com/resource/en/datasheet/stm32f410cb.pdf