IACR Transactions on Cryptographic Hardware and Embedded Systems
ISSN 2569-2925, Vol. 2022, No. 3, pp. 165–191. DOI:10.46586/tches.v2022.i3.165-191

The Wiretap Channel for Capacitive PUF-Based Security Enclosures

Kathrin Garb, Marvin Xhemrishi, Ludwig Kürzinger and Christoph Frisch

Technical University of Munich, Munich, Germany
firstname.lastname@tum.de, chris.frisch@tum.de

Abstract. In order to protect devices from physical manipulations, protective security enclosures were developed. However, these *battery-backed* solutions come with a reduced lifetime, and have to be actively and continuously monitored.
In order to overcome these drawbacks, *batteryless* capacitive enclosures based on Physical Unclonable Functions (PUFs) have been developed that generate a key-encryption-key (KEK) for decryption of the key chain. In order to reproduce the PUF-key reliably and to compensate the effect of noise and environmental influences, the key generation includes error correction codes. However, drilling attacks that aim at partially destroying the enclosure also alter the PUF-response and are subjected to the same error correction procedures. Correcting attack effects, however, is highly undesirable as it would destroy the security concept of the enclosure. In general, designing error correction codes such that they provide tamper-sensitivity to attacks, while still correcting noise and environmental effects is a challenging task.
We tackle this problem by first analyzing the behavior of the PUF-response under external influences and different post-processing parameters. From this, we derive a system model of the PUF-based enclosure, and construct a wiretap channel implementation from q-ary polar codes. We verify the obtained error correction scheme in a Monte Carlo simulation and demonstrate that our wiretap channel implementation achieves a physical layer security of 100 bits for 306 bits of entropy for the PUF-secret. Through this, we further develop capacitive PUF-based security enclosures and bring them one step closer to their commercial deployment.

Keywords: Physical Unclonable Function · Capacitive Enclosure · Error Correction · Wiretap Channel · Polar Codes · Physical Layer Security

1 Introduction

Devices deployed for high-security applications, such as Hardware Security Modules (HSMs) [BB19], have to withstand tampering attacks. For this purpose, cryptographic modules require a physical boundary in the form of enclosures or coatings [Fed08, ISO17, ISO12] that can detect a tamper event and trigger an alarm, zeroing all relevant Critical Security Parameters (CSPs) on the device.
The first generation of enclosures contained meshes of electrodes that continuously monitor resistivity to determine whether a tamper-event occurred [OI18, W.L, IMJFC13, Adv12]. However, monitoring required a continuous power supply in the form of batteries, which reduced the overall lifetime of the device and came with a higher sensitivity towards environmental influences.
To overcome these drawbacks, capacitive meshes based on Physical Unclonable Functions (PUFs) have been developed without the need for a continuous power supply [IOK+18, ION+19]. A PUF is an object's fingerprint that stems from manufacturing variations

Received: 2022-01-15 Accepted: 2022-03-15 Published: 2022-06-08

[Mae12], for instance, of electrodes that form the capacitive mesh within the enclosure. The capacitive differences within the mesh constitute the PUF-response, from which a Key-Encryption-Key (KEK) is generated to decrypt further CSPs of the key hierarchy [OHHS18, GOFK21]. Tampering attacks damage the enclosure and change electrode capacitance, which will destroy the PUF-key, and, thus, lead to a loss of all CSPs. However, also environmental changes affect the measured PUF-response and have to be compensated to a certain degree through error-correction codes.
This "dilemma" of ensuring tamper-sensitivity to attacks, while allowing environmental changes within a certain range is challenging. Not only does it require the integration of suitable error correction codes into the PUF post-processing, but what is more, a mechanism to model both cases separately. We address this problem through the following contributions.

Our contributions:

- We characterize the effects of temperature changes and tampering attacks on the PUF-based capacitive enclosure. Based on the obtained data, we discuss distortion and the impact of an attack on the PUF-based capacitive enclosure using appropriate quantization schemes.
- We introduce q-ary polar codes for error-correction of our Higher Order Alphabet PUF and apply a wiretap code on the practical data, obtained from measurements of the PUF-response, to protect the PUF-secret from an attacker. We also discuss the differences between our code and other wiretap codes for PUFs.
- We demonstrate how to achieve a physical-layer security level of up to 100 bits, for a PUF-secret of 306 bit length.

Overview:
In Section 2, we give an overview of tamper-sensitive PUFs, and relevant error correction codes. This is followed by an analysis of the PUF-response influenced by temperature changes and drilling attacks in Section 3. We include the effects of different choices of quantization, determining suitable parameters that model the capacitive enclosure. We also discuss different key generation schemes and their suitability in the context of the capacitive PUF-based security enclosure.
Section 4 proposes a solution to the described dilemma. Based on our system analysis in Section 3, we construct a wiretap channel through q-ary polar codes and verify our results in a Monte Carlo simulation. We conclude our work in Section 5.

2 Related Work

This section discusses tamper-sensitive PUFs and relevant error correction codes. We also highlight state-of-the-art approaches, where wiretap coding previously has been used for PUFs to improve the overall security.

2.1 Tamper-Sensitive Physical Unclonable Functions

The minuscule manufacturing variations of the PUF and physical unclonability make it suitable for tamper-sensitive applications.
One of the first larger scale PUF-based secure architectures is the optical waveguide polymer PUF [VNK+15, SFIC14], where a polymer waveguide covers the top area of a Printed Circuit Board (PCB). To detect whether a tamper event occurred, light from Light-Emitting Diodes (LEDs) is sent through the polymer waveguide, and the created light patterns are analyzed. A drawback of the optical waveguide polymer PUF is, however,

that it does not protect the whole device but only the top area of the PCB. Optical PUFs were also applied to smartcards by Esbach et al. [EFK+12].
Another electromagnetic PUF arranges antennas within an electromagnetic-sensitive sealing material [TZP20]. The wavelength of the radio signals varies due to manufacturing variations of the sealing material. The measurement of the channel state information between antennas determines whether a tamper event occurred. A similar approach is the anti-tamper radio proposed by Staat et al. [STZP21].
Smaller-scale PUFs coatings aim to protect small areas on integrated circuits (ICs). One such example was proposed by Tuyls et al. [TSŠ+06], where the capacitance of randomly arranged particles within a coating is measured in order to determine a tamper event.
Zhang et al. [ZHW+21] recently proposed Switched-Capacitor PUFs as protection of electrical circuits from manipulation.
Another technology closely related to challenge-response PUFs is a 3D hardware canary introduced by Briais et al. [BCC+12]. Here, a security-sensitive circuit is surrounded by a wire cage. A spatially distributed chain of functions placed at the vertices of the cage forms the hardware canary. A challenge has to be answered by a correct response to attest to the canary's integrity. The main difference between PUFs and hardware canaries is that hardware canaries are constructed through an algorithm, while PUFs are based on random manufacturing variations.
Capacitive PUF-based enclosures [IOK+18, ION+19] protect larger areas and entire PCBs, as discussed in more detail in Section 3.

2.2 Error Correction Codes

The application of Physical Unclonable Functions requires a sufficient level of reliability under environmental changes. For this purpose, PUF-key generation includes an error correction step. There are various error correction codes of PUFs that focus on optimizing the code in terms of implementation overhead, decoding complexity and hardware resources [BGS+08, MVHV12, HKS20, MHK+19, PMB+15] applying Bose Chaudhuri Hocquenghem (BCH), Reed Solomon or Reed Muller Codes. Another way of reducing the implementation overhead is by including soft and reliability information [MTV09a, MTV09b, MPB18, HOSB16].
Apart from implementation efficiency, secrecy leakage is a known issue for PUFs. Hence, several schemes were proposed to tackle this problem, focusing either on information leakage due to bias in the PUF-response [MLSW16, BY19, IHL+19] or on the reduction of helper data leakage [CW19, BY21]. Müelich and Bossert circumvented the leakage problem by proposing a novel secure sketch, where no additional helper data is required [MB17]. One of the main goals of the error correction code is to optimize the failure probability. This was the objective of Chen et al., who applied Polar Codes to SRAM PUFs [CIW+17].
In the context of PUF-based enclosures, achieving tamper-sensitivity is a significant issue, as we will see in the subsequent section. An attempt to ensure high-sensitivity to attacks while still correcting environmental effects was proposed by Immler and Uppund, who applied Limited Magnitude Codes (LMCs) [IU19], where only errors of a certain magnitude are corrected.

2.3 Wiretap Coding and PUFs

The task of an error correction code for PUFs is to target the noise effects in the PUF-response to derive an error-free secret key. Wiretap codes, on the other hand, have an additional feature: not only can they correct errors in the noisy PUF-response, but they also incorporate a security aspect. Wyner has shown in [Wyn75] that additional randomness besides the error correction capability is necessary to achieve security. Consequently, not

every error correction code applied to PUFs is a suitable wiretap code. Hence, several papers have studied wiretap codes, in particular for PUFs. They have been introduced for PUFs in [HÖ17] for codes of up to length 64 bit. In [BY19] and [BY21] the authors extend this preceding work to larger binary polar codes. However, in comparison to the proposed approach in this work, there are three major differences: (i) In the state-of-the-art applications of wiretap coding for PUFs, the goal is debiasing instead of physical layer security. (ii) The attacker in the state-of-the-art approaches is assumed to be weaker than in this work: In [HÖ17, BY19, BY21], the attacker receives the helper data as output of his channel. Hence the whole PUF-response is interpreted as error induced over the wiretapper's channel. In this work, the received message is degraded in comparison to the legitimate channel only based on the impact of a drilling attack. (iii) We operate on q-ary instead of binary polar codes; the enclosure PUF in this work allows for a higher-order quantization, and thus, more entropy can be extracted from the PUF than in a binary case. Consequently, we propose a novel scheme because any PUF error correction cannot detect an attacker, and the previous wiretap codes for PUFs do not match our attacker model and the non-binary response of our enclosure PUF. This is detailed in the following sections.

3 Key Generation for Capacitive PUF-Based Enclosures

The capacitive PUF-based enclosure requires several blocks: The measurement of the PUF-response (3.1), processing of the analog PUF-data before quantization (3.2), quantizing analog data to symbols (3.3), and finally, error-correction on these symbols including key generation (3.4).
Previous work on the capacitive enclosure, including an overview of the system components, is summarized in Subsection 3.1. An overview of post-processing steps, as proposed in [ION+19, IOK+18], is given in 3.2.1.
To enhance the existing PUF-model, we provide a detailed analysis of the impact of temperature effects and drilling attacks on the enclosure in Section 3.2. Our analysis is based on the measured PUF-responses obtained from [ION+19]. In Section 3.3, we analyze the distortion and error for different types of quantization. The goal of our analysis is to obtain an estimate for the error probability of our channel model and to investigate binary and q-ary channel models. Section 3.4 focuses on the different possible key generation schemes for the PUF-model and q-ary channels.

3.1 The Capacitive Enclosure

The 18.5 cm × 9 cm capacitive envelope (B-TREPID) [IOK+18] consists of two layers, each with 16 copper (Cu) electrodes (Rx and Tx) — arranged in a meander structure — that are separated by an insulating layer of polyimide (PI). The electrodes have a width and distance of 100 µm, as depicted in Figure 1. To reduce the impact of alternating electric fields on the capacitive measurement, the top and bottom of the envelope were extended by an additional shielding layer of Cu [Obe19, IOK+18], leading to an overall thickness of approximately 0.3 mm. Together with a cable for external communication, the envelope is wrapped around a casing containing the protected PCB, depicted in Figure 2.
The Cu electrodes overlap and form 16×16 *absolute* capacitances, which are, however, not suitable for the evaluation as a PUF since they depend on global manufacturing variations. The actual PUF-response is, hence, formed by measuring the difference between these absolute values, resulting in 128 *differential* capacitances with an estimated maximum entropy of 560 bits. These differential capacitances are obtained by measuring two Tx-electrodes (Tx pair) against one Rx-electrode. From the PUF-response, a secret KEK is derived (key enrollment) and repeatedly reproduced (key reproduction). A drilling attack

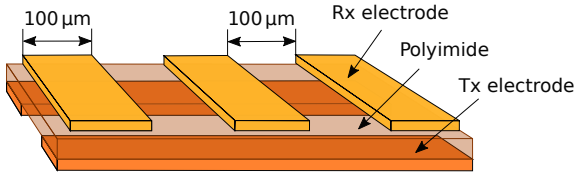


Figure 1: Schematic cross section of the envelope mesh with overlapping Rx and Tx electrodes.

with a diameter of 300 µm destroyes two of the electrodes, leading to a loss of 80 bits of entropy [IOK+18] in the unprocessed PUF-response, which alters the reproduced key.
To measure the PUF-response, a discrete measurement circuit was developed [OIHS18]. Since this circuit was optimized in terms of accuracy and not size, a smaller measurement IC of approximately 5 mm × 5 mm was developed and integrated into the envelope [FIU+18]. The IC measures 16 differential capacitances in parallel, which decreases the measurement time. However, the optimization in terms of space and time comes at the expense of accuracy.
A microcontroller carries out the post-processing of the PUF-response on the PCB running the Embedded Key Management System (EKMS) [OHHS18, GOFK21]. The EKMS is a hardened FreeRTOS with integrated key management and key generation. This also includes the error correction algorithm to compensate for noise and environmental effects.
To cover only parts of a device, an enclosure (COVER) was developed, which differs from the envelope (B-TREPID) in its use case and mesh arrangement. COVER, in contrast to the envelope, is not wrapped around the device but only covers the top or bottom area of the PCB.
We measured and statistically analyzed the PUF-distribution of 50 envelopes (B-TREPID) [IOK+18] and 115 enclosures (COVER) [ION+19]. The differential capacitances of B-TREPID and COVER are both Gaussian distributed and behave equivalently under external influences. However, their overall range in femtofarads differs, which makes merging both data sets difficult. In general, the distribution of COVER is broader compared to B-TREPID. Since the COVER data is more extensive than the B-TREPID data, the following analysis will focus on data obtained from the COVER.

3.2 Analysis of the PUF-Response

This section covers an analysis of the measured capacitances and how they are affected by temperature changes and drilling attacks.

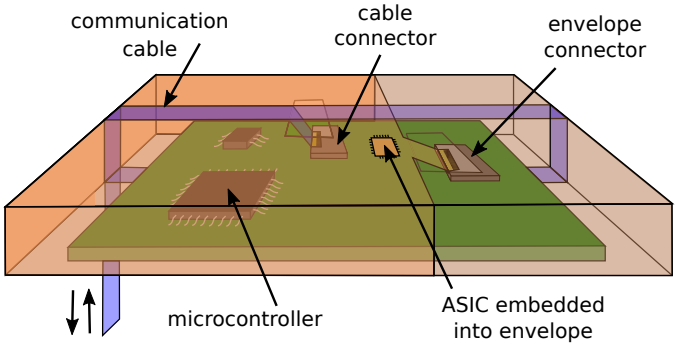


Figure 2: Schematic overview of the capacitive envelope protecting a PCB.

3.2.1 General Overview

For the following analysis, the PUF-response was measured with the discrete circuit [OIHS18] instead of the IC [FIU+18] to increase accuracy. Furthermore, since more data is available for COVER [ION+19] compared to the B-TREPID [IOK+18] and since both systems behave equivalently, the PUF-responses for the following analysis stem from COVER.
The 128 unprocessed differential capacitances are integers in the interval $[-10000, +10000]$ that are Gaussian distributed with a standard deviation of $\sigma = 30\,\text{fF}$ (corresponding to 2241 points) after normalization (see Post-Processing) measured over 115 COVERs [ION+19]. The $[-10000, +10000]$ interval corresponds to a full-scale range of $[-134\,\text{fF}, +134\,\text{fF}]$ with a digital resolution of 13.4 aF corresponding to 1 point [Obe19]. Measuring a single differential capacitance — also called "node" — with the discrete circuit takes 390 µs on average, while the differential measurement time for the entire enclosure amounts to 50 ms.

Noise Distribution
Despite low-noise components and noise filtering within the measurement circuit, an inevitable noise caused by the full setup of the enclosure and the measurement system still remains. The noise plays a major role in the quantization of the PUF-response, which is discussed in Subsection 3.3. The noise distribution was statistically determined from 200 consecutive measurements of the same COVER. It is well-approximated through a Gaussian with a standard deviation of 1.7 fF corresponding to 129 points [Obe19] centered around $\mu = 0\,\text{fF}$.

Post-Processing
After measuring the PUF-response, several post-processing steps are necessary before the generation (or reproduction) of the PUF-key. Figure 3 depicts the three steps of post-processing with analog helper data generation:

1. Shift of TX groups (normalization)
2. Generation of quantization mapping
3. Generation of analog helper data

The raw differential capacitances are obtained by measuring two Tx electrodes against one Rx electrode. We will refer to each of these Tx-pairs measured against all other Rx electrodes as a Tx group. However, *global* manufacturing variations in the thickness of the electrodes lead to offsets in the raw differential capacitance of some of the Tx groups [OIHS18]. In order to remove the dependency on these global variations of each Tx group, we subtract the group mean in the first step. We will refer to the Tx group shift as "normalization".

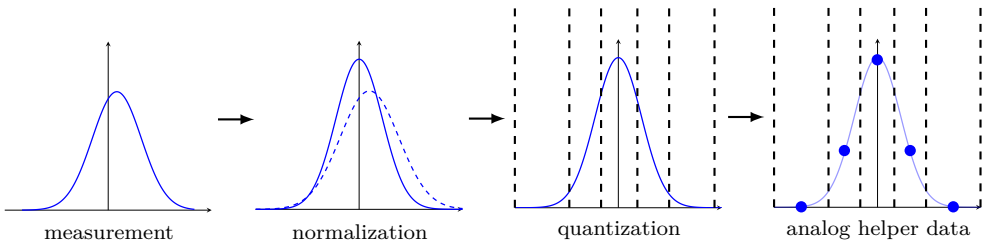


Figure 3: Post-processing of the PUF-response after measurement and helper data generation.

In the second step, the PUF-response is quantized, hence, it is divided into quantization intervals. To reduce the quantization error, the PUF-values of each interval are shifted to the center of the interval (step 3), which further reduces the quantization error (see [IHKS16]). This offset is stored as *analog helper data*. Even though, this step reduces the quantization error, it is not sufficient to compensate all errors stemming from noise, let alone environmental changes, as we will see in the following. Subsection 3.3 will discuss quantization in more detail.

3.2.2 Temperature Distribution

Environmental effects, for example, changes in temperature and humidity, alter the measured PUF-response. The effects of electromagnetic interference were considered in the design of the PUF and the measurement circuit. A copper shielding was added to the envelope to counteract the effect of alternating electric fields. Without the shielding, reliably measuring the differential capacitances is not possible. The effects of external magnetic fields are counteracted by the narrow excitation frequency and canceled out through the meander structure of the PUF. Hence, for the following analysis, we focus solely on temperature effects. Fig. 4 shows the absolute capacitances for temperatures between $-20\,°\mathrm{C}$ and $60\,°\mathrm{C}$. The corresponding differential capacitances are depicted in Fig. 5a before normalization (Tx group shift) and in Fig. 5b after.

Apart from a shifted mean, the normalized distribution (Fig. 5b) also appears more narrow than the distribution of the raw capacitances (Fig. 5a). The normalized distribution exhibits a lower variance, as all Tx group means are shifted separately.

Fig. 5c depicts the change in the measured differential capacitance compared to a reference distribution at $20\,°\mathrm{C}$. The maximum difference amounts to 1500 points at $60\,°\mathrm{C}$ (Fig. 5d), where most node changes are within $[-700, +700]$ points. Fig. 5e and Fig. 5f depict histograms of the differential capacitance for temperatures between $0\,°\mathrm{C}$ and $60\,°\mathrm{C}$ before and after normalization. The Gaussian fits show that — in both cases — as the temperature increases, the distribution broadens, even though the standard deviation is of a smaller magnitude ($\Delta\sigma = 207$) after normalization. In the case of the raw distribution, the mean is shifted to the right as the temperature increases.

The enclosure protects devices that have to resist large environmental changes in the field that depend on the reliability requirements for the particular use case. For instance, the security policy for the HP Atalla Ax160 PCI HSM [Hew] states that for temperatures

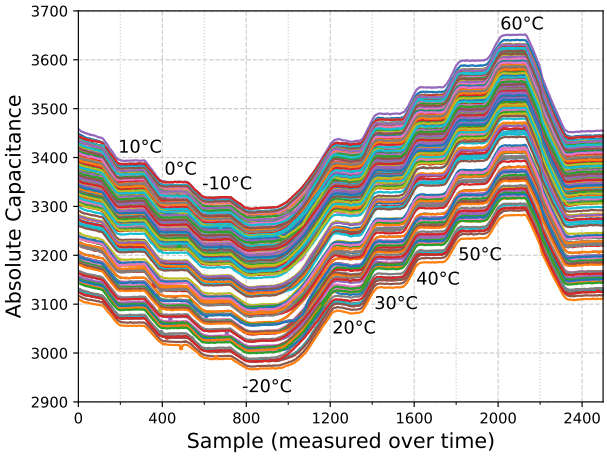
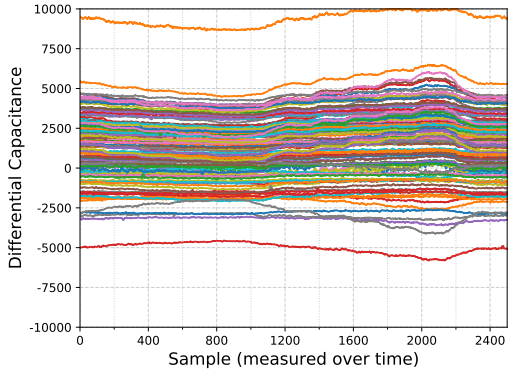
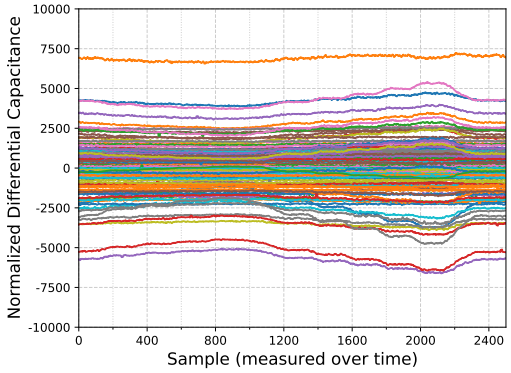


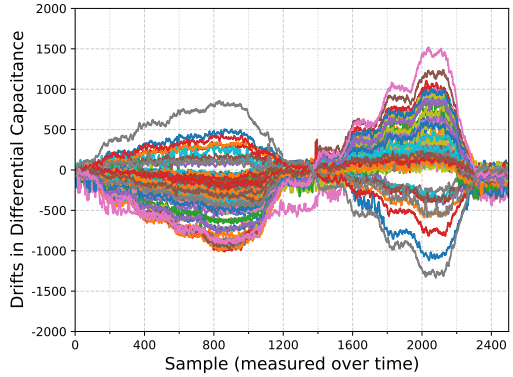
Figure 4: Absolute capacitance under temperature change. Data taken from [ION+19].



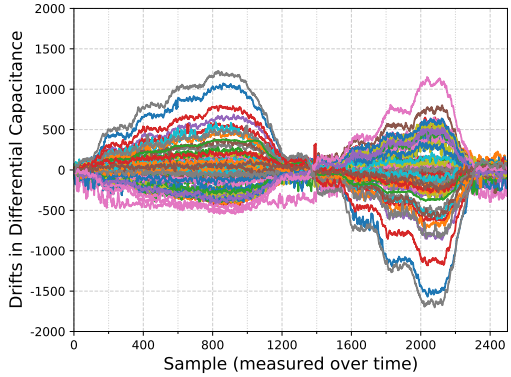
(a) Raw differential capacitance under temperature change (compare to Fig. 4). Data taken from [ION+19].



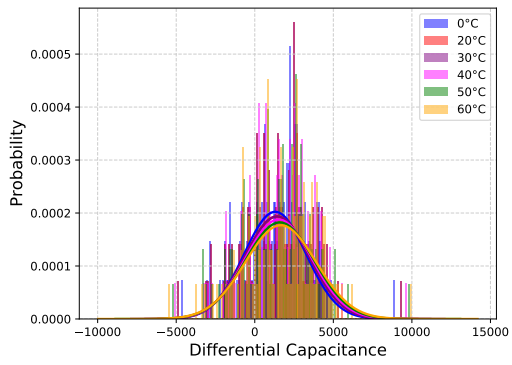
(b) Differential capacitance under temperature change *after* Tx group shift.



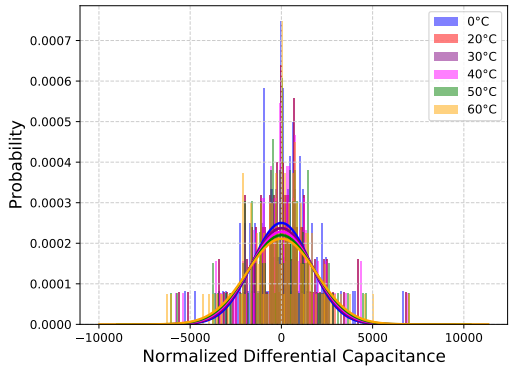
(c) Change in differential capacitance due to temperature changes compared to reference measurement (at 20 °C) *before* Tx group shift.



(d) Change in differential capacitance due to temperature changes compared to reference measurement (at 20 °C) *after* Tx group shift.



(e) Distribution of the differential capacitance (with Gaussian fit) for temperatures between 0 °C and 60 °C *before* Tx group shift.



(f) Distribution of the differential capacitance (with Gaussian fit) for temperatures between 0 °C and 60 °C *after* Tx group shift.

Figure 5: Analysis of the differential capacitance under temperature changes.

outside of the range $[-20\,°\text{C}, 100\,°\text{C}]$, a tamper-event is generated. Hence, an envelope or COVER protecting such a device will have to withstand temperature changes outside of the interval $[-20\,°\text{C}, 60\,°\text{C}]$. Larger temperatures create even larger offsets in the raw

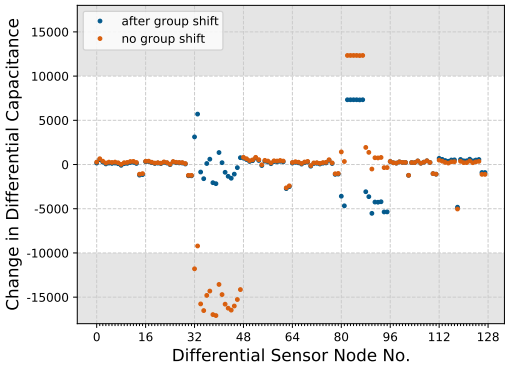


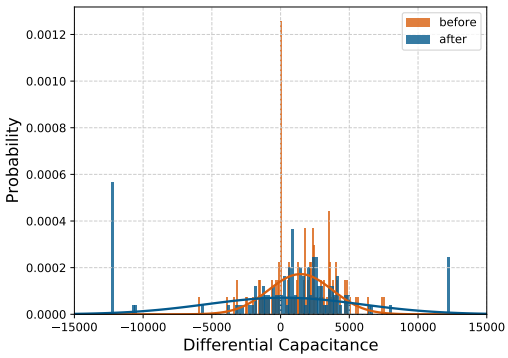
Figure 6: Change in differential capacitance for a drilling attack affecting two Tx groups (before and after normalization). The grey area denotes values within the range of non-linear effects.

differential data and, thus, broaden the distribution.

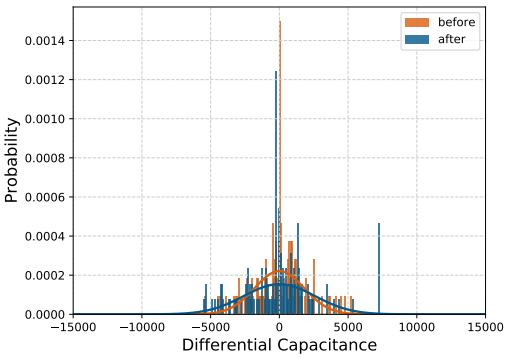
3.2.3 Attack Distribution

The envelope and COVER are designed to withstand drilling attacks with a diameter of 300 µm since both the width and distance of the electrodes are 100 µm. Hence, a drilling attack destroys at least two electrodes.

As we will see, drilling attacks affect the enclosure in two different ways. Fig. 6 shows the change in the PUF-response due to such a drilling attack before and after normalization. The normalization reduces the large offsets and shifts points outside of the $[-10000, +10000]$ back to the the distribution center. Values outside of the ± 10000 full-scale range (highlighted in grey) are subjected to non-linear effects (clipping) during the measurement. Since, in general, values outside of that range are also possible during a regular measurement, a simple check for values outside of the full-scale range is not sufficient to determine an attack. Furthermore, attacks with smaller drilling diameters might lead to more minor changes in the differential capacitance. The changes in Figure 6 show that the attack "muddles up" the differential capacitances of the affected Tx groups.



(a) Raw distribution of the differential capacitance (with Gaussian fit) before and after the attack.



(b) Normalized distribution of the differential capacitance (with Gaussian fit) before and after the attack.

Figure 7: Analysis of the effect of a drilling attack on the differential capacitance.

These burst errors have to be considered in the error correction.
Apart from burst errors, drilling attacks also lead to a broadening of the PUF-distribution. Fig. 7a and 7b depict the histograms of the PUF-response before and after the attack with and without normalization. Before the normalization (Fig. 7a), the distribution broadens significantly with a change in standard deviation by 3295 points. In the case of normalization, the broadening of the distribution is reduced to 787 points since through an attack, a specific Tx group will be more affected than others. However, it still significantly exceeds the broadening due to temperature changes.

3.3 Quantization of the PUF-Response

Quantization is an essential step in post-processing the PUF-response, as discussed in Subsection 3.2. In the following, we discuss previous quantization schemes that were analyzed in the context of the capacitive PUF-based envelope. Based on the analysis of the PUF in the previous subsections, we determine how different choices of quantization intervals affect the PUF-response.

3.3.1 Previous Work on Quantization

Before analyzing the choice of quantization intervals in more detail, we first give an overview of previous quantization schemes discussed in the context of the PUF-based security enclosure. Note that, the quantization does not signify the mapping from continuous to discrete capacitive values in this case. However, it represents an additional step performed on the already discretized PUF-response — after the Analog-to-Digital Converter (ADC) — to reduce the impact of noise during a regular measurement.
Previous discussions on quantization schemes — in the the envelope context — focused on the advantages and disadvantages of equidistant versus equiprobable quantization intervals [IHKS16, IHL+19, IU19, ION+19]. Equiprobable intervals lead to a uniform distribution of the normalized capacitances, while equidistant intervals distribute the capacitances unevenly. In the case of equiprobable quantization, the analog helper data, stored to shift the PUF-data to the middle of the interval, might leak information about the location of the PUF-data within the distribution. However, since the helper data reside within the enclosure boundary, they are difficult to access.
Apart from this comparison, previous work also comprised variable-length quantization with equidistant intervals. This approach maps the PUF-response to binary values of variable length in order to optimize the per-bit minimum entropy [IHL+19]. The authors tailored the variable-length mapping to Varshamov-Tenengolts (VT) codes, performing a single insertion, deletion, or substitution.
However, variable-length codes are susceptible to increased error propagation in case critical bits or symbols are lost; this may require synchronization correction and, thus, complicates the decoding process [GN98, CRR98]. Another drawback is that a single PUF-node might be distributed over several intervals, which makes its behavior in terms of external influences, like attacks or environmental changes, less predictable.
The choice of an optimal quantization scheme is a non-trivial task, especially for noisy sources [GN98]. In the following, we aim at providing more clarity on the quantization behavior by investigating the effect of noise and external influences.

3.3.2 Quantization and Distortion

The quantization of the PUF-response follows two goals. On the one hand, the choice of quantization intervals determines the entropy of the PUF-key, and hence, should yield sufficient accuracy. On the other hand, the intervals should be chosen such that the quantized PUF-response is reliable enough, with a decreased susceptibility to noise. The

optimization of both goals is mutually exclusive and comes with a particular trade-off. Apart from this trade-off, the quantization also determines the leakage of the PUF-response. Equidistant intervals leak information about the PUF through varying symbol probabilities, while equiprobable quantization leaks information through the analog helper data. Hence, in both cases, we accept an unavoidable information leakage.

The quantization quality can be measured as distortion [GN98] and describes how well the original variable x can be reproduced through the quantization $\hat{x}$. In the following, we consider a simple distortion measure d, the Mean Squared Error (MSE), $d(x, \hat{x}) = (x - \hat{x})^2$, which for a sequence of length n reads:

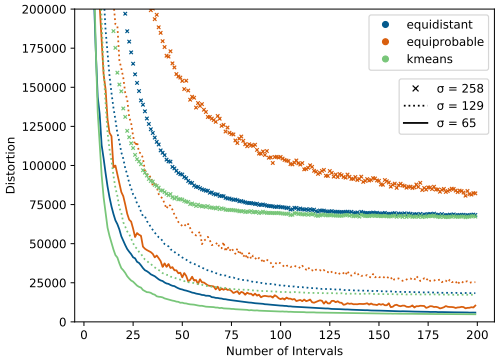
$$d(x^n, \hat{x}^n) = \frac{1}{n} \sum_{i=1}^{n} d(x_i, \hat{x}_i) = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{x}_i)^2$$

We simulated the distortion for a set of 1000 noisy PUF-responses, as depicted in Figure 8a, for equidistant (blue), equiprobable (red), and k-means (green) quantization from a total of 5 to 200 intervals. We see that the distortion will not decrease further at a certain number of intervals. This threshold depends on the amount of noise ($\sigma_n = 65$, $\sigma_n = 129$, and $\sigma_n = 258$) that occurs during a regular PUF-measurement.

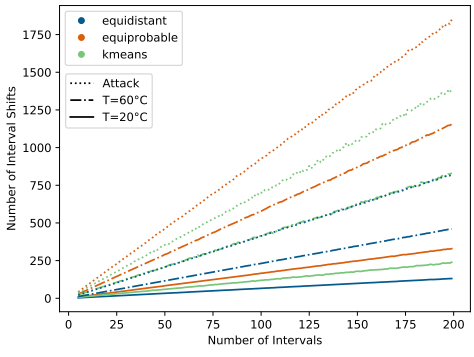
The number of interval shifts for a PUF-response is depicted in Figure 8b. We see that, in general, equidistant quantization has a lower probability of interval shifts occuring. The reason for this is that the innermost intervals of an equiprobable (or k-means) quantization are very narrow, and hence, more minor offsets will have a higher impact compared to equidistant quantization. We also see that the difference in interval changes between the three cases ($T = 20°$, $T = 60°$, attack) will grow with an increasing number of intervals.

Quantized values should be encoded such that the distance between adjacent intervals is minimized. For binary numbers, this could be represented by a Gray code mapping. Figure 9 depicts the Bit Error Ratio (BER) for Gray code mapping comparing PUF measurements at $T = 20°$, and $T = 60°$, against a drill attack. We see that — for a number of intervals ≤ 64 — the BER at an elevated temperature can be distinguished from the BER of a drill attack. However, for a large number of quantization intervals, both curves approach the same BER.

This is due to the Gray Code mapping, where each interval is represented by a binary number of $\log_2(m)$ bits, where m is the number of intervals. Within the adjacent $\log_2(m)$



(a) Distortion for equidistant (blue), equiprobable (red), and k-means (green) quantization with three different noise distributions, $\sigma_n = 65$, $\sigma_n = 129$, and $\sigma_n = 258$ for 5 to 200 intervals.



(b) Number of interval shifts in a PUF-response at $T = 20°$, $T = 60°$, and drilling attacks for equidistant (blue), equiprobable (red), and kmeans (green) quantization.

Figure 8: Distortion (a) and number of changed intervals (b) for equidistant (blue), equiprobable (red), and kmeans (green) quantization.

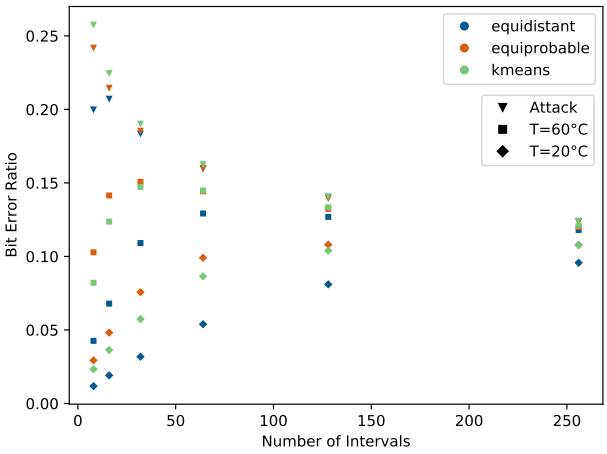


Figure 9: Bit Error Ratio for Gray Code mapping comparing temperatures $T = 20°$ and $T = 60°$ against a drilling attack.

intervals, each interval change — that corresponds to a bit flip — increases the Hamming distance of the encoded value by one. However, when the corresponding PUF-node is changed by more than $\log_2(m)$ intervals, the Hamming distance decreases again. Figure 9 depicts the bit error rates for Gray encoding with different quantization schemes and temperatures.
Previous assumptions led to a choice of 40 equidistant quantization intervals with a width of 500 points [IHKS16]. This corresponds to $3.9\sigma_n$, where 99.99% of values from the noise distribution are within the confidence interval. In general, we see that equidistant quantization leads to a lower BER and a smaller number of interval changes.
For 40 intervals, we see that the distortion has not yet reached the threshold, and the difference in interval shifts is still relatively small. The distortion for equidistant intervals and k-means reaches the threshold at approximately 100 intervals; however, the BER for Gray encoding decreases significantly from 64 intervals to 128. A similar behavior can be observed for non-local and balanced Gray Codes and the binary representation of consecutive decimal numbers. Codes with a larger number of bits, for instance, Johnson code, lead to an even faster decrease of the BER due to a large number of bits per symbol. Hence, these binary mappings do not describe the error behavior of Figure 8b accurately enough, and what is more, even benefit the attacker when the number of intervals increases. In general, modeling the enclosure through a binary channel is difficult. Thus, we focus on a q-ary channel model for our wiretap channel implementation in Section 4.
Quantization represents the final step of post-processing. The quantized and encoded PUF-response is the input to the key generation, which is discussed in the following. Note that although we employ a wiretap code to increase security, we still utilize schemes for error correction.

3.4 Generating Keys from Physical Unclonable Functions

In the following, we give an overview of key generation schemes and discuss their suitability for the capacitive PUF-based enclosure.

3.4.1 Key Generation Schemes

When it comes to choosing a key generation scheme, one can either consider pointer-based schemes, where additional reliability information about the PUF is included, or linear

schemes, where a key is generated without considering properties of the PUF.
In the first case, the advantage of additional reliability comes at the expense of discarding certain PUF-bits. Applying pointer-based methods to the enclosure leads to "blind spots"; hence, only reliable parts of the PUF-response constitute the PUF-key. Thus, the PUF-key becomes less susceptible to changes through an attack, making pointer-based schemes [YD10, HMSS12, HWRL+13, YHD15] unsuitable for the capacitive enclosure.
A variety of linear schemes do not consider reliability information about the PUF and, hence, derive a key based on the complete PUF-response. One of the first schemes proposed for PUFs, the Fuzzy Commitment [JW99], which was extended by adding the quantization and post-processing steps for the capacitive enclosure, is depicted in Figure 10.
It consists of two major steps: (i) During the enrollment at the manufacturer, a key is chosen and the corresponding helper data are generated; to be more precise, a true random number R generates the secret S, which is then encoded to the codeword C. (ii) In the reproduction phase, the key is repeatedly restored through measuring the PUF-response during the regular operation of the protected device.
In the enrollment phase, the measured and normalized PUF-response X is quantized $q(x)$, and the analog helper data W' are generated and stored in NVM. From the codeword together with the quantized PUF-response $\widehat{X}$, the helper data W are calculated and stored in NVM to end the enrollment. The "channel" is obtained by repeatedly reproducing the PUF-reponse that is subjected to noise ϵ_n, environmental effects like temperature changes ϵ_t, and possible drilling attacks ϵ_a. This can be seen as a "faulty" codeword "transmitted" over a noisy channel, that when decoded during reconstruction, yields the secret S. In order to verify the secret, which is a KEK, the key chain is decrypted. If this is successful, the secret S is valid. Otherwise, the alarm and zeroization are triggered. [1]
The Fuzzy Extractor [DRS04] is another scheme, which is similar to the Fuzzy Commitment. In this case, however, the PUF-response and not a true random number constitutes the secret, which is why the secret is hashed in order to reduce helper data leakage.
A scheme, where no random number from a True Random Number Generator (TRNG) is generated, is the Syndrome Construction [DRS04, DORS08]. Just as in case of the Fuzzy Extractor, the secret is equivalent to the PUF-response, which makes additional hashing necessary.
In the basic Syndrome Construction, the helper data $W = \widehat{X}\, H^{\mathrm{T}}$ are defined via the parity check matrix H. The reconstruction consists of minimizing the error e in $W = (\widehat{X} + e)\, H^{\mathrm{T}}$. In [CIW+17]. This approach is implemented in the context of polar codes, however, as we will see in Section 4 it is not suitable for wiretap coding.
Another scheme that does not require an additional random number R is Systematic Low Leakage Coding (SLLC) [HYP15]. In this case, the PUF-response $X = X_{\mathrm{S}} + X_{\mathrm{M}}$ is split into the secret X_{S} and a part X_{M} for masking. The helper data is defined as $W = X_{\mathrm{S}} P \oplus X_{\mathrm{M}}$. SLLC has the drawback that hashing is required. Furthermore, it can only be applied to systematic codes, where the information and redundancy are separated.
A key generation scheme based on the secure sketch that does not require additional helper data was proposed by Müelich and Bossert [MB17]. The Helper Data Algorithm (HDA) is necessary to create a codeword from the PUF-response, however, Müelich and Bossert constructed the code, such that the PUF-response corresponds to a codeword. A drawback that the authors described was the increased complexity of their scheme.
For the capacitive PUF-based enclosure, we choose the Fuzzy Commitment as depicted in Figure 10 due to two main reasons: (i) This allows for a full flexibility regarding the key. Because the key is generated from a TRNG, a second enrollment is possible e.g. after transport (see [GOFK21]). Besides, no hash function is required. (ii) As discussed in

[1] The Fuzzy Commitment scheme is not limited to binary linear codes only, but e.g. Reed Solomon codes (e.g. with $q-$ary symbols) are also possible [CS16]. This also extends to the $q-$ary polar code in Section 4.2.1.

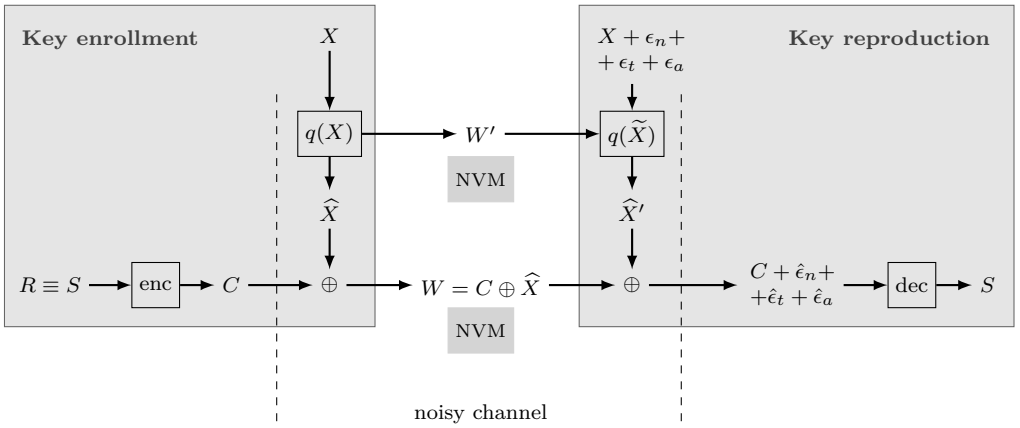


Figure 10: Key generation via Fuzzy Commitment including post-processing and quantization.

Section 4.2.2, it is necessary for the wiretap setting to introduce randomness. This is not possible for key generation schemes such as Syndrome constructions or SLLC because the key and helper data only depend on the PUF response without additional randomness.

As the overall key generation scheme has been chosen, the next step is to find a suitable encoder and decoder for the error correction.

3.4.2 Previous Error Correction Codes

Several error correction codes were proposed for the enclosure, for instance, Reed Solomon, BCH, or VT codes [IU19, IHL+19]. However, a major issue regarding error correction in the context of the enclosure is that the chosen code might correct changes in the PUF-response that originate from an attack. Many of the previously proposed schemes do not address this problem.

To tackle this issue, error correction based on Limited Magnitude Codes (LMCs) was proposed [IU19, JL12], where only the Least Significant Bits (LSBs) of each of the PUF-symbols are corrected. Through this, the error correction is restricted to the neighboring intervals, leaving larger offsets caused by an attack unchanged. In general, LMCs provide a simple method to incorporate tamper-sensitivity without considering complex error patterns.

In order to describe the more complex patterns of burst errors and changes in the PUF-distribution, we propose a wiretap channel implementation based on polar codes. By choosing Polar Codes, we are able to model both effects — environmental influences and attacks — separately, as we will see in the next section. This separation requires a wiretap code and is not feasible with regular PUF error correction codes.

4 Implementing the Wiretap Channel

The applied error correction code has to be able to distinguish between environmental changes and an attack. A theoretical model that incorporates this separation between two different channels, is the wiretap channel [Wyn75, OW85, CK06]. In the following, we model the error behavior of the enclosure with the help of the wiretap channel, and propose an implementation based on polar codes.

4.1 The Wiretap Channel and Attacker Model

The wiretap channel originally regards the problem of a wiretapper eavesdropping on a discrete, memoryless channel [Wyn75]. Wyner showed that a reliable transmission with a finite capacity is possible, while achieving approximately perfect secrecy. For this, he assumed that the wiretapper eavesdrops the transmission via a second channel. Hence, the goal is to provide a reliable transmission on the main channel, hiding it from the second channel accessed by the wiretapper. In general, there exist different assumptions for error probability and secrecy [OW85].

The separation into a legitimate and a noisy channel can be applied to the capacitive PUF-based enclosure in order to improve the error correction. We adapt the wiretap channel for our purposes, see Figure 11, such that on the main channel, the codeword $C = \widehat{X} \oplus W$ is "transmitted" with error probability p_1, considering changes in the PUF-distribution that stem from noise $\hat{\epsilon}_n$ and temperature changes $\hat{\epsilon}_t$. On the second channel, the codeword is additionally affected by changes due to an attack $\hat{\epsilon}_a$, which leads to a different error probability p_2.

Previous polar code implementations of wiretap channels in the PUF-context focused on binary silicon PUFs and hiding secrecy leakage from unstable or biased PUF-bits [HÖ17, BY19, BY21]. These wiretap codes dissected their implementation into a regular channel and a channel where the distorted helper data is transmitted. Our use case and code construction significantly differ from previous implementations since the helper data do not have any relevance in our case. We construct the wiretap channel from a channel modeling the regular transmission of the PUF-response, while the other channel models the PUF-response under attack, e.g., through a drilling attack. What is more, our code construction is based on q-ary polar codes. To the best of our knowledge, we are, hence, the first to provide a wiretap channel design and implementation with higher-order alphabet PUFs targeting physical layer security.

Regarding $\hat{\epsilon}_a$, we define our attacker to have two possibilities: (i) When the overall device enclosed by the PUF is powered off, the attacker can remove the envelope. He can then try to decrypt the data that has been secured by the PUF key. (ii) During runtime of the device, he can drill a hole into the envelope and try to gain access to the sensitive data in a minimally invasive way to prevent zeroization.

An investigation of micro-drilling attacks, magnetic probing, and bypassing of electrodes was published by Garb et al. [GSHO21] in the context of the capacitive PUF-based enclosure, including countermeasures. The enclosure was designed [ION+19, IOK+18] to withstand 250-300 µm diameter drilling attacks that can be reliably detected. Smaller holes are achieved through laser beams with high aspect ratios. Focused Ion Beams (FIBs)

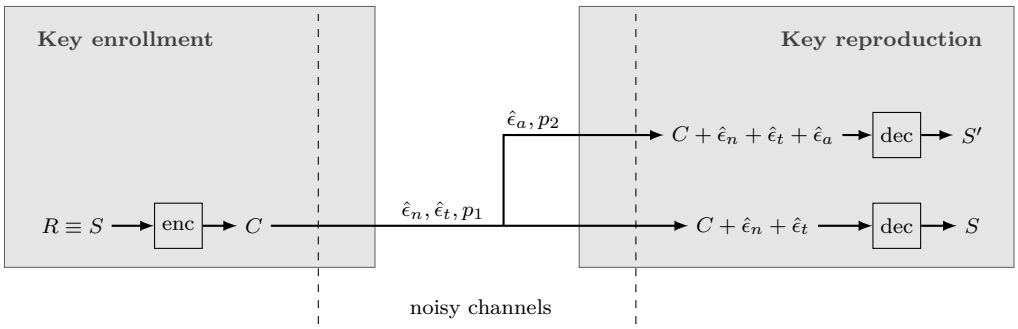


Figure 11: A wiretap channel adaptation for measurement noise $\hat{\epsilon}_n$, environmental changes $\hat{\epsilon}_t$ and drilling attacks $\hat{\epsilon}_a$.

are, in general, more suited for processing surface structures. An attempt to drill a deep hole with FIBs will lead to debris at the bottom of the hole due to the removed material, making small holes with a diameter of approximately 10 µm infeasible. However, with lasers and high aspect ratios, small holes in the two-digit micrometer range are, in general, achievable. The thickness of the casing and an additional potting material hinder an attacker from reaching the components on the PCB since a deeper hole comes with a larger diameter. Another factor is the probing of critical components, which is limited by the shaft width of the probing needle. The tip length of the smallest commercially available probing needles amounts to 3.3-5 mm, while the shaft width is usually in the higher two-digit micrometer range and will, hence, not fit through an arbitrarily small hole. The reach of the probing needle can be further limited by increasing the casing thickness. Since small holes, in general, require additional countermeasures that can not be achieved through error correction, we focus on detecting holes with a diameter of 250-300 µm. The envelope provides good protection against fault injection since it covers the entire device. The only component that can be externally accessed is the voltage supply. However, voltage glitching is mainly hampered by monitoring the voltage supply. Also, hardware-based countermeasures through an optimized choice of electronic components make voltage glitching attacks practically infeasible. When switching out the PUF-based solution against a battery-backed system, the power supply will reside within the boundary of the enclosure, leaving no attack point for Laser Fault Injection (LFI) or glitching attacks. However, even in this case, faults induced by radiation are still conceivable and have to be counteracted by extensive software hardening.

In the following, we provide an implementation of the wiretap channel through q-ary polar codes such that in either attack scenario, (i) and (ii), security is achieved. Our code construction can be applied to other single-challenge PUFs in the context of error correction. In our system model, the PUF serves as a KEK that protects the other critical security parameters in the key chain. Also, device authentication can be achieved by creating additional key pairs [GOFK21].

4.2 Polar Codes

Polar codes were introduced by Stolte [Sto03] and later reintroduced by Arıkan [Ari09]. Arıkan showed that polar codes achieve the capacity of a binary-input discrete memoryless channel (DMC) under successive cancellation (SC) decoding. Later works extended Arıkan's result to polar codes defined over a binary extension field, i.e., polar codes with symbols from the extension field $\mathbb{F}_q$, where q is a power of two. Yuan and Steiner introduced the construction of polar codes using a kernel defined over an extension field [YS18]. Besides, polar codes also provide a capacity-achieving construction for the wiretap channel with symmetric component channels [MV11].

Unlike previous polar codes for PUFs [HÖ17, BY19, BY21], we focus on a generalized q-ary polar code to extract more entropy and a longer secret from the PUF. Our analysis in Subsection 3.3 showed that modeling the PUF-response with a binary symmetric channel led to small changes in the PUF-response from a tampering attempt. Thus, to use a valid wiretap coding scenario that guarantees a higher security level, we need to shift to a higher quantization level. The fact that the PUF-response is analog motivates even further the usage of a quantization process with more than just two levels. The advantages of a q-ary model for the enclosure are an increased sensitivity towards tampering and a higher extraction of entropy. In the following, we discuss the encoder and decoder for the capacitive PUF-based enclosure and the code construction.

4.2.1 Encoder and Decoder

Figure 12 defines a polar code over a finite field $\mathbb{F}_q$ [YS18]. Similarly to binary polar codes [Ari09], q-ary polar codes are based on a process called *channel polarization*. Let the source output be two $q-$ary symbols (u_1, u_2) and let their encoded version be two $q-$ary symbols (c_1, c_2). Then the relation between those two vectors is

$$(c_1, c_2) = (u_1, u_2) \cdot \begin{pmatrix} 1 & 0 \\ \alpha & 1 \end{pmatrix} \triangleq (u_1, u_2) \cdot \mathrm{F}_2(\alpha) \tag{1}$$

where α is an optimization parameter [YS18], $\mathrm{F}_2(\alpha)$ is the polarization matrix, which in the $q-$ary case depends on α. As illustrated in Figure 12, two copies of the same physical channel (defined by the channel law $P(y|c)$) are polarized into two virtual channels. The first channel with an input u_1 and an output (y_1, y_2) has a lower rate than the physical channel, and the second channel that inputs u_2 and outputs (y_1, y_2, u_1) has a higher rate than the physical channel [Ari09]. This case can be generalized for a code length $n > 2$, as depicted in Figure 13. The information symbols $u = (u_1, u_2, u_3, u_4, \ldots, u_n)$ are encoded using a polar transform that yields the codeword $c = (c_1, c_2, c_3, c_4, \ldots, c_n)$. The relation between u and c can be written as

$$c = u \cdot \mathrm{F}_2(\alpha)^{\otimes \log_2 n} \tag{2}$$

where $\mathrm{F}_2(\alpha)^{\otimes \log_2 n}$ denotes the $\log_2 n-$fold Kronecker product of the matrix $\mathrm{F}_2(\alpha)$ with itself. The channel output is denoted by $y = (y_1, y_2, y_3, y_4, \ldots, y_n)$, which represents the erroneous observation of c. Similarly, n physical channels will be polarized into channels with a rate higher than the physical one, and into channels with a lower rate than the physical channel [Ari09]. Arıkan showed that with $n \to \infty$, the channel polarization results in either perfect or useless channels, i.e., channels with capacity 1 or 0. These results were also later generalized to the $q-$ary case [PB12]. However, for short or moderate code length, the channel polarization is far from the asymptotic one, thus, allowing some channels to be considered as *mediocre*, i.e., neither perfect nor useless. The aim of the code construction is to find the best polarized channels and transmit information over those and freeze the other channels, i.e., transmit a predetermined *frozen* value over the bad channel that contains no information. We will use the symbol 0 as a frozen value and we denote the frozen positions by the set $\mathcal{F}$.

The decoder is depicted in Figure 14. The task of the decoder is to determine an estimate $\hat{u} = (\hat{u}_1, \hat{u}_2, \hat{u}_3, \hat{u}_4, \ldots, \hat{u}_n)$ of the source vector, knowing the received y and the frozen positions $\mathcal{F}$ and their values. SC decoding is the original decoding algorithm that Arikan proposed for polar codes under which they achieve capacity [Ari09].

SC decoding is a decoding strategy that allows the use of soft information and can be implemented recursively due to the structure of polar codes. The main idea is that the soft information is propagated from the right to the left (see Figure 14) and *check node* operations are performed (depicted with an XOR operation and iterated as $\tilde{c}_i$ in Figure 14). Then after the soft information has reached the very top left positions (the positions

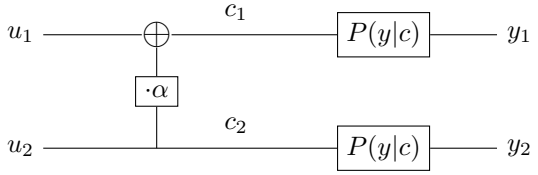


Figure 12: Polar code 2×2 kernel defined over a $q-$ary field. Note that α is an element of the field $\mathbb{F}_q$.

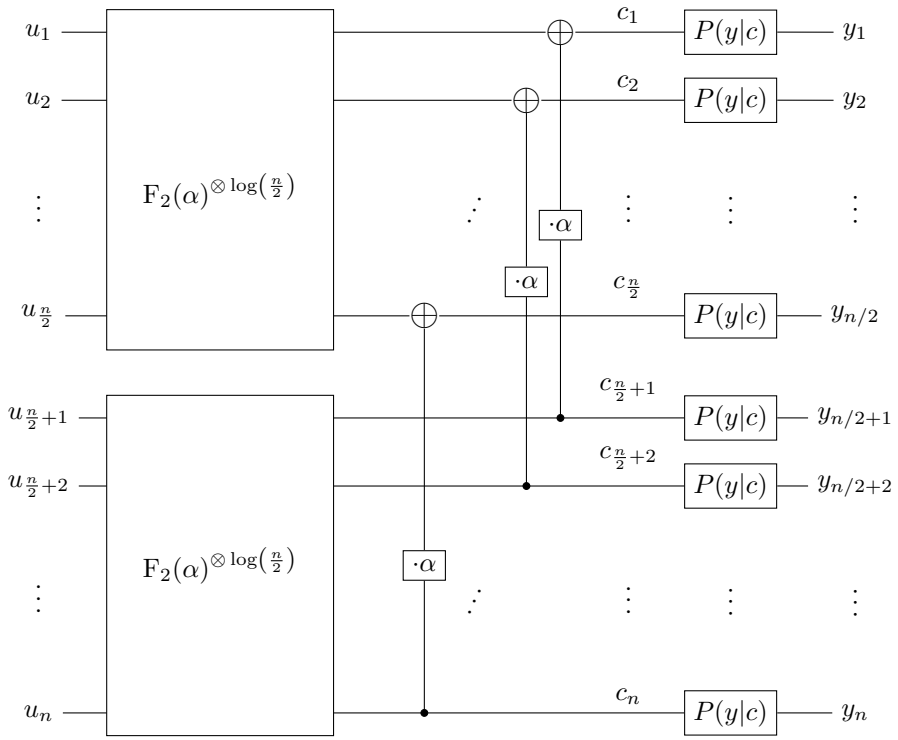


Figure 13: Polar encoder

of $\hat{u} = (\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_n)$ in Figure 14), a hard decision must be performed by taking into consideration the soft information available. Then, the hard decision is propagated from left to right , thus, allowing the decoder to successively decide on the bits that were most likely transmitted. The hard decision is then "mixed" with the soft information for the *variable node* operation (depicted with a bullet point and called v_i in Figure 14).
Moreover, since the decoder knows the positions and the values of the frozen bits, it can bypass the hard estimation for those particular positions and "decide" on the correct information. However, hard decisions can mislead to a miscorrected symbol, thus allowing errors to be propagated for later hard decisions. This issue was solved by Tal and Vardy in their work in [TV11], where they introduced successive cancellation list (SCL) decoding. SCL decoding allows the decoder to pursue many choices for an information position once it needs to make a hard decision for a non-frozen symbol. However, to allow a practically feasible decoder, a certain list size L was introduced. Once the list of paths has been filled up (when L possible paths are stored in the list stack), the decoder needs to discard the most unlikely paths and only pursue the most likely ones. The work in [TV11] showed that even the smallest possible $L = 2$ decoder improved the performance compared to SC decoding and almost maximum likelihood decoding performance was achieved by a relatively small list size $L = 32$ for some parameters [TV11]. For the $q-$ary case, the authors in [YS18] proposed a pruned SCL decoder that only pursues the paths that are within a threshold δ of reliability. The main difference to conventional SCL decoding is that the list does not have to be full to start pruning. The performance is very comparable to the conventional SCL decoding, but the number of operations is significantly reduced in the case where transmission occurs over a good channel [YS18].
Tal and Vardy [TV11] showed that indeed polar codes suffer from a bad weight distribution compared to other state-of-the-art codes since many times the transmitted codeword was part of the final decoding list but was not the most likely to be picked. Thus, a modification

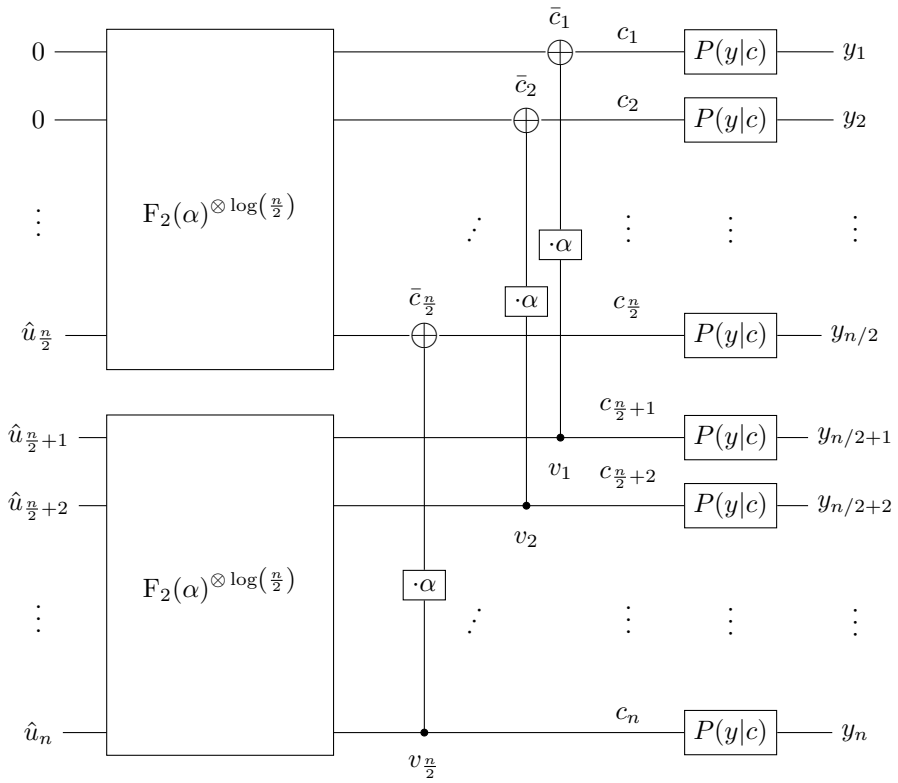


Figure 14: Polar decoder

of the decoding is needed such that it picks — with a high probability — the transmitted codeword from the final list (if it is a member of it). This can be achieved by a code concatenation with CRC [TV11] or by deploying a hash computation of all the candidates of the final list [CIW+17].

4.2.2 Code Construction

The aim of our code construction is to determine the polarized channels that are "bad" for the attacker channel and "good" for the legitimate channel affected by noise and temperature changes. We estimate the quality of the polarized channels by using the Monte-Carlo code construction [Ari09] for both types of channels (legitimate and eavesdropper). However, since we will operate on a short length regime (the length of the polar code n is much smaller than infinity), we cannot have fully polarized symbol channels. In [MV11], a polar code construction is presented that achieves the secrecy capacity of a wiretap channel when the channel components are symmetric. This polar code construction allows the polarization of the symbol-channels $u_i \leftarrow y_1, y_2, \ldots, y_n, u_1, \ldots u_{i-1}$ for the channels of the legitimate receiver and the eavesdropper. Then, the polarized symbol-channels that are "very bad" are frozen, i.e., the input symbol is set to a fixed value, the polarized symbol-channels that are "good" for the legitimate receiver but "bad" for the eavesdropper are used to transmit information symbols and finally, the polarized symbol channels that are "good" for both receivers are used to transmit random symbols that carry no information. In order to estimate the quality of the polarized symbol-channels we need a realistic channel model for both the legitimate receiver and attacker (eavesdropper). In general, at least one Tx and one Rx electrode will be destroyed through an attack with a 250 to 300 µm drill. As explained in Section 3, the PUF-response is analog, and a natural

solution is to quantize it. A temperature change or a tampering attempt, as shown in Subsection 3.2, can change the value of the PUF-response. Thus, the changed value may fall into another quantization level other than the original one. Therefore, this effect can be modeled using a q-ary channel with non-zero crossover probabilities. We simulated the probability matrix for all possible symbol changes with 100,000 PUF-responses drawn from the PUF-distribution considering the changes in the PUF-response as modeled in Section 3. We model the legitimate and attacker channel through q-ary channels for 8, 16, and 32 equiprobable intervals. In contrast to equidistant quantization, equiprobable intervals provide the required uniformly distributed input symbols at the polar encoder. A bias in the distribution of the input symbols can lead to additional leakage that has to be taken into consideration [CW19]. We observed that the channel for the legitimate receiver is not symmetric and therefore every $P(y_i|c_j) = p_{i,j}$ can be different for $i, j \in \{0, 1, \ldots, q-1\}$. We obtained the $p_{i,j}$ from simulating 100,000 PUF-responses for the legitimate and the attacker channel with the system model parameters described in Section 3. The simulation was done under consideration of the PUF-post-processing steps with and without analog helper data.

Since our PUF consists of 8 Tx pairs and 16 Rx electrodes, its response will contain 128 values quantized into 128 q-ary symbols. We use a code-offset solution for error correction and, thus, need to construct a polar code of length $n = 128$. Note that the length of the polar code is independent of the value of the quantization level q. We construct the Polar Code through a Monte Carlo simulation [Ari09]. The core idea of the Monte Carlo code construction is to use uncoded transmission (encode n symbols into a codeword of n symbols) and to add noise to the codeword according to the channel model. Then, SC decoding is performed over the received vector using the soft information of the channel model. The decoding is done as usual, except that after a hard decision, the decoder checks if the decision was correct or not, counts the occurrence of incorrect decisions, and — if necessary — changes the decision to the correct symbol. The idea behind this construction method is to estimate the probability that the decoding of a symbol $\hat{u}_i$ is wrong given correct previous hard decisions $(\hat{u}_1, \ldots, \hat{u}_{i-1}) = (u_1, \ldots, u_{i-1})$. If we ran this algorithm infinitely many times, the estimated probabilities would be the real ones, but a sufficiently good estimation can be achieved with finitely many runs. We perform this estimation method for both channels, namely the legitimate and the attacker one. We freeze the channels that are very bad (high probability of an erroneous decision) for both channels. We also transmit random symbols for the channels that are very good (probability of an erroneous decision ≈ 0) and use all others for transmitting information symbols. We then calculate the entropy in bits for the attacker, by taking into account the estimations from the Monte Carlo construction. For every polarized channel symbol where information is transmitted, we check the distribution of a SC decoding. We checked that many outcomes were uniformly distributed among the extension field (ensuring maximum entropy), and some outcomes were leaning a bit more towards the correct symbol. However, the fact that we can ensure uniformly distributed input symbols (due to equiprobable quantization) does not allow the attacker to guess using the probability of each symbol (MAP decoding). An entropy of s bits means that the attacker can at its best case brute-force 2^s options for the decoding process. For SCL decoding this means that the list size L should be at least $L > 2^s$ large in order to have the correct codeword in the final list. Due to the code-offset construction, we add the PUF-response to a codeword of the polar code generated randomly and store the helper data.

The results from the Monte Carlo code construction are listed in Table 1 for three different numbers of intervals q, with and without analog helper data W'. Legitimate symbol channels with a probability for incorrect decoding below the threshold d are selected to estimate the entropy of the attacker channel H_{att} in bits. Hence, this threshold is also a measure of the overall reliability of the legitimate channel. The selection results in a

Table 1: The q-ary polar code construction results for 8, 16, and 32 intervals. The codeword consists of 128 symbols. Here, d denotes the per-symbol error probability that indicates the error correction reliability. The number of reliable symbols after error correction is given as n_s, of which n_f symbols are readable for an attacker. The attacker entropy H_{att} denotes the achievable security level from the remaining secret symbols in an attack scenario. We distinguish between decoding with and without analog helper data W'.

	Without W'					With W'				
q	d	n_{s}	n_{f}	H_{att}	H_{secret}	d	n_{s}	n_{f}	H_{att}	H_{secret}
8	0.0500	91	11	113.5	273	0.0500	123	22	163.0	369
	0.0100	85	11	95.6	255	0.0100	121	22	157.9	363
	0.0050	82	11	86.8	246	0.0050	120	22	154.9	360
	0.0010	73	11	60.8	219	0.0010	119	22	151.9	357
	0.0005	71	11	55.6	213	0.0005	117	22	145.9	351
	0.0001	65	11	40.2	195	0.0001	112	22	130.9	336
	-					10^{-5}	106	22	112.0	318
	10^{-6}	56	11	22.1	168	**10^{-6}**	**102**	**22**	**100.3**	**306**
	-					$< 10^{-6}$	101	22	98.0	303
16	0.0500	76	9	110.0	304	0.0500	99	17	158.9	396
	0.0100	70	9	87.5	280	0.0100	91	17	127.1	364
	0.0050	67	9	76.3	268	0.0050	88	17	115.3	352
	0.0010	61	9	55.7	244	0.0010	82	17	92	328
	0.0005	58	9	47.8	232	0.0005	79	17	80.8	316
	0.0001	52	9	33.1	208	0.0001	74	17	63.9	296
	-					10^{-5}	68	17	45.6	272
	10^{-6}	45	9	15	180	10^{-6}	64	17	34.9	256
	-					$< 10^{-6}$	63	17	32.0	252
32	0.0500	80	11	168.7	400	0.0500	86	15	181.0	430
	0.0100	75	11	143.0	375	0.0100	78	15	141.4	390
	0.0050	72	11	129.4	360	0.0050	76	15	131.7	380
	0.0010	68	11	111.4	340	0.0010	73	15	116.9	365
	0.0005	66	11	102.5	330	0.0005	72	15	112.4	360
	0.0001	62	11	89.1	310	0.0001	69	15	98.9	345
	-					10^{-5}	62	15	74.0	310
	10^{-6}	**55**	**11**	**57.3**	**275**	10^{-6}	58	15	58.9	290
	-					$< 10^{-6}$	56	15	49.9	280

number of symbols that are good for the legitimate channel n_{s}. This scheme can reliably reproduce n_s symbols containing an entropy of $H_{\text{secret}} = n_s \cdot \log_2(q)$ bits. The dimension of the code is $k = n_s$ in this wiretap construction. H_{secret} corresponds to the bit length of the secret in conventional PUF key scenarios without wiretap secrecy leakage. These bits can be directly used as a symmetrical encryption key or hashed to fit into a certain key length.

Of the reliable symbols n_{s}, n_{f} symbols are also good for the attacker channel and have to be randomized. An attacker receives the remaining $n_s - n_f$ symbols exhibiting a high error rate from noisy to entirely random. The complexity for an attacker is expressed through $H_{\text{att}} = -\sum_i^{n_s} p_{s,i} \log_2(p_{s,i})$, where $p_{s,i}$ denotes the symbol error rate after an attack. H_{att} determines the physical layer security level of the wiretap PUF scheme.

As the parameter d determines the number of symbols n_s, this parameter relates to the security level and enables a trade-off between security and reliability; a stricter threshold d selects only the most reliable bits, but also reduces the number of legitimate bits n_s for the secret and the complexity for an attacker H_{att}.

As the first Monte Carlo simulation only yielded per-symbol error rates, we performed

Table 2: Decoding results for T=20° using the q-ary Polar codes. The number of correctly transmitted symbols is denoted as n_s. Of those n_s symbols, n_f symbols are readable in an attack scenario, while the remaining $n_s - n_f$ symbols exhibit a high error rate. With that, we estimate the secret length in bits as H_{secret} and the complexity of an attacker to H_{att}.

Decoder	q	W'	FER	n_{s}	n_f	H_{att}	H_{secret}
SCD	8	yes	4.0×10^{-6}	102	22	100	306
SCL ($L = 8$)	8	yes	1.0×10^{-6}	102	22	100	306
SCD	32	no	7.0×10^{-6}	55	11	57	275
SCL ($L = 8$)	32	no	3.3×10^{-6}	55	11	57	275

another Monte Carlo simulation to obtain the Frame Error Rate (FER) for specific code construction parameters of Table 1 with the SC and SCL decoder. The simulation results are listed in Table 2 for a temperature of $T = 20°$. The FER is the probability of the PUF secret being wrongly decoded.
The results show that we obtain a PUF-secret with up to 306 bit length with 8 intervals, and 275 bits for 32 intervals, while reaching a FER in the order of 10^{-6}. The entropy of the attacker channel amounts to 100 bits for 8 intervals. For 32 intervals, the brute force effort for the attacker is reduced to 2^{57}, while still preserving 275 bits of entropy for the PUF-secret.
The polar code design supports temperature changes in the range $[+5°, +35°]$, which is compatible with the operating temperatures of state-of-the-art network HSMs [Gro, Hew]. Polar codes are praised for their low encoding and decoding complexity, especially for the binary case. The encoding and the SC decoding can be performed in $\mathcal{O}(n \log n)$ time, while SCL decoding is performed in $\mathcal{O}(L \cdot n \log n)$ [TV11, BSPB15]. As for the $q-$ary polar codes, the check node and variable operations require $\mathcal{O}(q \log q)$ and $\mathcal{O}(q)$, respectively, instead of $\mathcal{O}(1)$ for the binary case. For a low number of intervals, the decoding complexity for q-ary codes reaches a low number of additional operations, which makes even q-ary polar codes suitable for implementation on a microcontroller.

5 Conclusion

In this paper, we constructed a wiretap channel for the capacitive PUF-based enclosure from q-ary polar codes, and modeled the effects of attacks on the PUF-response.
First, we analyzed how temperature changes and drilling attacks affect the PUF-distribution based on real data obtained from measurements of the PUF-response. From this analysis, we derived a system model of the enclosure considering the impact of the PUF post-processing and different choices of quantization intervals.
To construct the polar code for our Higher Order Alphabet PUF, we modeled the error behavior of the capacitive PUF-based enclosure through q-ary channels, and selected the best symbol channels for the legitimate wiretap channel, while minimizing the good symbol channels for the attacker. The wiretap code protects the information stored in these symbols, a code property that non-wiretap codes could not achieve.
With a Monte Carlo simulation for 8, 16, and 32 intervals and two different decoders, we demonstrated a physical layer security of 100 bits, while preserving 306 bits of entropy for the PUF-secret.

Acknowledgment

The authors would like to thank DSO National Laboratories, Singapore, for publication permission of the measured and processed PUF-responses, and Fraunhofer AISEC for supporting the data acquisition. Thanks to Peihong Yuan for providing us with good

insights about the implementation. Thanks also to Michal Virgovič for plotting scripts, and to Georg Maringer, Hedongliang Liu, and Matthias Hiller for valuable discussions. This work was partly funded by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (Grant Agreement No. 801434), and by the German Ministry of Education and Research in the project VE-FIDES under grant number 16ME0257.

References

[Adv12] Advanced Cryptographic Hardware Development IBM Poughkeepsie and IBM Research. *IBM 4765 cryptographic coprocessor security modulesecurity policy.* Zürich, December 2012.

[Ari09] Erdal Arikan. Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transactions on information Theory*, 55(7):3051–3073, 2009.

[BB19] Elaine Barker and William C. Barker. *Recommendation for Key Management: Part 2 – Rev. 1 – Best Practices for Key Management Organizations.* National Institute of Standards and Technology, Gaithersburg, MD, May 2019.

[BCC+12] Sébastien Briais, Stéphane Caron, Jean-Michel Cioranesco, Jean-Luc Danger, Sylvain Guilley, Jacques-Henri Jourdan, Arthur Milchior, David Naccache, and Thibault Porteboeuf. 3d hardware canaries. In Emmanuel Prouff and Patrick Schaumont, editors, *Cryptographic Hardware and Embedded Systems – CHES 2012*, pages 1–22, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[BGS+08] Christoph Bösch, Jorge Guajardo, Ahmad-Reza Sadeghi, Jamshid Shokrollahi, and Pim Tuyls. Efficient helper data key extractor on fpgas. In *IACR CHES*, pages 181–197. Springer Berlin Heidelberg, 2008.

[BSPB15] Alexios Balatsoukas-Stimming, Mani Bastani Parizi, and Andreas Burg. Llr-based successive cancellation list decoding of polar codes. *IEEE Transactions on Signal Processing*, 63(19):5165–5179, 2015.

[BY19] Yonghong Bai and Zhiyuan Yan. A secure and robust key generation method using physical unclonable functions and polar codes. In *2019 IEEE International Workshop on Signal Processing Systems (SiPS)*, pages 254–259, 2019.

[BY21] Yonghong Bai and Zhiyuan Yan. A secure and robust puf-based key generation with wiretap polar coset codes. *Journal of Electronic Testing*, 37, 06 2021.

[CIW+17] Bin Chen, Tanya Ignatenko, Frans M. J. Willems, Roel Maes, Erik van der Sluis, and Georgios Selimis. A robust sram-puf key generation scheme based on polar codes. In *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pages 1–6, 2017.

[CK06] I. Csiszar and J. Korner. Broadcast channels with confidential messages. *IEEE Trans. Inf. Theor.*, 24(3):339–348, sep 2006.

[CRR98] R. Chandramouli, N. Ranganathan, and S.J. Ramadoss. Adaptive quantization and fast error-resilient entropy coding for image transmission. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(4):411–421, 1998.

[CS16] Sonam Chauhan and Ajay Sharma. Fuzzy commitment scheme based on reed solomon codes. In *Proceedings of the 9th International Conference on Security of Information and Networks*, pages 96–99, 2016.

[CW19] Bin Chen and Frans M. J. Willems. Secret key generation over biased physical unclonable functions with polar codes. *IEEE Internet of Things Journal*, 6(1):435–445, 2019.

[DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM journal on computing*, 38(1):97–139, 2008.

[DRS04] Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *IACR Eurocrypt*, pages 523–540. Springer, 2004.

[EFK+12] Thomas Esbach, Walter Fumy, Olga Kulikovska, Dominik Merli, Dieter Schuster, and Frederic Stumpf. A new security architecture for smartcards utilizing pufs. In *Information Security Solutions Europe (ISSE 2012)*. Vieweg Verlag, 2012.

[Fed08] Federal Office for Information Security. Common Criteria Protection Profile – Cryptographic Modules, Security Level "Enhanced" (BSI-CC-PP-0045), 2008.

[FIU+18] E. Ferres, V. Immler, A. Utz, A. Stanitzki, R. Lerch, and R. Kokozinski. Capacitive multi-channel security sensor ic for tamper-resistant enclosures. In *IEEE SENSORS*, pages 1–4, 2018.

[GN98] R.M. Gray and D.L. Neuhoff. Quantization. *IEEE Transactions on Information Theory*, 44(6):2325–2383, 1998.

[GOFK21] Kathrin Garb, Johannes Obermaier, Elischa Ferres, and Martin König. Fortress: Fortified tamper-resistant envelope with embedded security sensor. In *18th Annual International Conference on Privacy, Security and Trust (PST2021)*, 2021.

[Gro] Thales Group. High assurance hardware security modules. https://cpl.thalesgroup.com/encryption/hardware-security-modules/network-hsms. Online, accessed on 16 January 2022.

[GSHO21] Kathrin Garb, Marc Schink, Matthias Hiller, and Johannes Obermaier. Attacks and countermeasures for capacitive puf-based security enclosures. In *2021 IEEE Physical Assurance and Inspection of Electronics (PAINE)*, pages 1–8, 2021.

[Hew] Hewlett-Packard Company. Hewlett-Packard — Atalla Security Products Ax160 PCI HSM Security Policy. https://www.pcisecuritystandards.org/ptsdocs/HP%20Atalla%20Ax160%20PCI%20HSM%20Security%20Policy%201_1.pdf. Online, accessed 14 November 2021.

[HKS20] M. Hiller, Ludwig Kurzinger, and G. Sigl. Review of error correction for pufs and evaluation on state-of-the-art fpgas. *Journal of Cryptographic Engineering*, 10:229–247, 2020.

[HMSS12] Matthias Hiller, Dominik Merli, Frederic Stumpf, and Georg Sigl. Complementary ibs: Application specific error correction for pufs. In *IEEE HOST*, pages 1–6. IEEE, 2012.

[HÖ17] Matthias Hiller and Aysun Gurur Önalan. Hiding secrecy leakage in leaky helper data. In *International Conference on Cryptographic Hardware and Embedded Systems*, pages 601–619. Springer, 2017.

[HOSB16] Matthias Hiller, Aysun Gurur Önalan, Georg Sigl, and Martin Bossert. Online reliability testing for puf key derivation. In *TrustED*, page 15–22, New York, NY, USA, 2016. Association for Computing Machinery.

[HWRL+13] Matthias Hiller, Michael Weiner, Leandro Rodrigues Lima, Maximilian Birkner, and Georg Sigl. Breaking through fixed puf block limitations with differential sequence coding and convolutional codes. In *TrustED*, pages 43–54, 2013.

[HYP15] Matthias Hiller, Meng-Day Yu, and Michael Pehl. Systematic low leakage coding for physical unclonable functions. In *ASIA CCS*, pages 155–166, 2015.

[IHKS16] Vincent Immler, Maxim Hennig, Ludwig Kürzinger, and Georg Sigl. Practical aspects of quantization and tamper-sensitivity for physically obfuscated keys. In *ACM CS2*, pages 13–18, 2016.

[IHL+19] Vincent Immler, Matthias Hiller, Qinzhi Liu, Andreas Lenz, and Antonia Wachter-Zeh. Variable-length bit mapping and error-correcting codes for higher-order alphabet pufs—extended version. *Journal of Hardware and Systems Security*, 3(1):78–93, 2019.

[IMJFC13] Phil Isaacs, Thomas Morris Jr, Michael J Fisher, and Keith Cuthbert. Tamper proof, tamper evident encryption technology. In *Pan Pacific Symposium*, 2013.

[IOK+18] Vincent Immler, Johannes Obermaier, Martin König, Matthias Hiller, and Georg Sigl. B-trepid: batteryless tamper-resistant envelope with a puf and integrity detection. In *IEEE HOST*, pages 49–56. IEEE, 2018.

[ION+19] Vincent Immler, Johannes Obermaier, Kuan Kuan Ng, Fei Xiang Ke, JinYu Lee, Yak Peng Lim, Wei Koon Oh, Keng Hoong Wee, and Georg Sigl. Secure physical enclosures from covers with tamper-resistance. *IACR CHES*, pages 51–96, 2019.

[ISO12] ISO, Geneva, Switzerland. *ISO/IEC 19790:2012 Information technology – Security techniques – Test requirements for cryptographic modules*, August 2012.

[ISO17] ISO, Geneva, Switzerland. *ISO/IEC 24759:2017 Information technology – Security techniques – Test requirements for cryptographic modules*, March 2017.

[IU19] Vincent Immler and Karthik Uppund. New insights to key derivation for tamper-evident physical unclonable functions. *IACR CHES*, pages 30–65, 2019.

[JL12] Myeongwoon Jeon and Jungwoo Lee. On codes correcting bidirectional limited-magnitude errors for flash memories. In *IEEE ISITA*, pages 96–100, 2012.

[JW99] Ari Juels and Martin Wattenberg. A fuzzy commitment scheme. In *Proceedings of the 6th ACM Conference on Computer and Communications Security*, CCS '99, page 28–36, New York, NY, USA, 1999. Association for Computing Machinery.

[Mae12] Roel Maes. *Physically unclonable functions: Constructions, properties and applications*. PhD thesis, Katholieke Universiteit Leuven, Belgium, 2012.

[MB17] Sven Mueelich and Martin Bossert. A new error correction scheme for physical unclonable functions. In *SCC 2017; 11th International ITG Conference on Systems, Communications and Coding*, pages 1–6, 2017.

[MHK+19] H. Mandry, A. Herkle, L. Kürzinger, S. Müelich, J. Becker, R. F. H. Fischer, and M. Ortmanns. Modular puf coding chain with high-speed reed-muller decoder. In *IEEE ISCAS*, pages 1–5, 2019.

[MLSW16] Roel Maes, Vincent Leest, Erik Sluis, and Frans Willems. Secure key generation from biased pufs: extended version. *Journal of Cryptographic Engineering*, 6, 06 2016.

[MPB18] Sven Müelich, Sven Puchinger, and Martin Bossert. Using convolutional codes for key extraction in sram physical unclonable functions, 2018.

[MTV09a] R. Maes, P. Tuyls, and I. Verbauwhede. A soft decision helper data algorithm for sram pufs. In *IEEE ISIT*, pages 2101–2105, 2009.

[MTV09b] Roel Maes, Pim Tuyls, and Ingrid Verbauwhede. Low-overhead implementation of a soft decision helper data algorithm for sram pufs. In *International workshop on cryptographic hardware and embedded systems*, pages 332–347. Springer, 2009.

[MV11] Hessam Mahdavifar and Alexander Vardy. Achieving the secrecy capacity of wiretap channels using polar codes. *IEEE Transactions on Information Theory*, 57(10):6428–6443, 2011.

[MVHV12] Roel Maes, Anthony Van Herrewege, and Ingrid Verbauwhede. Pufky: A fully functional puf-based cryptographic key generator. In Emmanuel Prouff and Patrick Schaumont, editors, *IACR CHES*, pages 302–319, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[Obe19] Johannes Obermaier. *Breaking and Restoring Embedded System Security*. PhD thesis, Technische Universität München, 2019.

[OHHS18] Johannes Obermaier, Florian Hauschild, Matthias Hiller, and Georg Sigl. An embedded key management system for puf-based security enclosures. In *MECO*, pages 1–6. IEEE, 2018.

[OI18] Johannes Obermaier and Vincent Immler. The past, present, and future of physical security enclosures: from battery-backed monitoring to puf-based inherent security and beyond. *Journal of Hardware and Systems Security*, 2(4):289–296, 2018.

[OIHS18] Johannes Obermaier, Vincent Immler, Matthias Hiller, and Georg Sigl. A measurement system for capacitive puf-based security enclosures. In *DAC*, New York, NY, USA, 2018. Association for Computing Machinery.

[OW85] L. H. Ozarow and A. D. Wyner. Wire-tap channel ii. In Thomas Beth, Norbert Cot, and Ingemar Ingemarsson, editors, *Advances in Cryptology*, pages 33–50, Berlin, Heidelberg, 1985. Springer Berlin Heidelberg.

[PB12] Woomyoung Park and Alexander Barg. Polar codes for q-ary channels, $q = 2^r$. In *2012 IEEE International Symposium on Information Theory Proceedings*, pages 2142–2146, 2012.

[PMB+15] S. Puchinger, S. Mueelich, M. Bossert, M. Hiller, and G. Sigl. On error correction for physical unclonable functions. In *ITG SCC*, pages 1–6, 2015.

[SFIC14] Merrielle Spain, Benjamin Fuller, Kyle Ingols, and Robert Cunningham. Robust keys from physical unclonable functions. In *IEEE HOST*, pages 88–92. IEEE, 2014.

[Sto03] Norbert Stolte. *Recursive Codes with the Plotkin-Construction and Their Decoding*. PhD thesis, Technische Universität Darmstadt, 2003.

[STZP21] Paul Staat, Johannes Tobisch, Christian T. Zenger, and Christof Paar. Anti-tamper radio: System-level tamper detection for computing systems. *CoRR*, abs/2112.09014, 2021.

[TSŠ+06] Pim Tuyls, Geert-Jan Schrijen, Boris Škorić, Jan van Geloven, Nynke Verhaegh, and Rob Wolters. Read-proof hardware from protective coatings. In *IACR CHES*, pages 369–383. Springer Berlin Heidelberg, 2006.

[TV11] Ido Tal and Alexander Vardy. List decoding of polar codes. In *2011 IEEE International Symposium on Information Theory Proceedings*, pages 1–5, 2011.

[TZP20] Johannes Tobisch, Christian Zenger, and Christof Paar. Electromagnetic enclosure PUF for tamper proofing commodity hardware and other applications. In *TRUDEVICE*, 2020.

[VNK+15] M. Vai, B. Nahill, J. Kramer, M. Geis, D. Utin, D. Whelihan, and R. Khazan. Secure architecture for embedded systems. In *IEEE HPEC*, pages 1–5, Sep. 2015.

[W.L] W.L. GORE & Associates, Inc., Electronic Products Division, Newark. *GORE™ Tamper Respondent Surface Enclosure*.

[Wyn75] A. D. Wyner. The wire-tap channel. *The Bell System Technical Journal*, 54(8):1355–1387, 1975.

[YD10] Meng-Day Yu and Srinivas Devadas. Secure and robust error correction for physical unclonable functions. *IEEE Design & Test of Computers*, 27(1):48–65, 2010.

[YHD15] Meng-Day Yu, Matthias Hiller, and Srinivas Devadas. Maximum-likelihood decoding of device-specific multi-bit symbols for reliable key generation. In *2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pages 38–43. IEEE, 2015.

[YS18] Peihong Yuan and Fabian Steiner. Construction and decoding algorithms for polar codes based on 2×2 non-binary kernels. In *2018 IEEE 10th International Symposium on Turbo Codes Iterative Information Processing (ISTC)*, pages 1–5, 2018.

[ZHW+21] Yin Zhang, Zhangqing He, Meilin Wan, Jiuyang Liu, Haoshang Gu, and Xuecheng Zou. A SC PUF standard cell used for key generation and anti-invasive-attack protection. *IEEE Transactions on Information Forensics and Security*, pages 1–1, 2021.