 doi:10.20944/preprints202011.0209.v1

Article

Secret Key Agreement with Physical Unclonable Functions: An Optimality Summary

Onur Günlü [1,*] and Rafael F. Schaefer [1]

[1] Information Theory and Applications Chair, Technische Universität Berlin; {guenlue, rafael.schaefer}@tu-berlin.de

* Correspondence: guenlue@tu-berlin.de; Tel.: +49-30-314-28463

Abstract: We address security and privacy problems for digital devices and biometrics from an information-theoretic optimality perspective, where a secret key is generated for authentication, identification, message encryption/decryption, or secure computations. A *physical unclonable function (PUF)* is a promising solution for local security in digital devices and this review gives the most relevant summary for information theorists, coding theorists, and signal processing community members who are interested in optimal PUF constructions. Low-complexity signal processing methods such as transform coding that are developed to make the information-theoretic analysis tractable are discussed. The optimal trade-offs between the secret-key, privacy-leakage, and storage rates for multiple PUF measurements are given. Proposed optimal code constructions that jointly design the vector quantizer and error-correction code parameters are listed. These constructions include modern and algebraic codes such as polar codes and convolutional codes, both of which can achieve small block-error probabilities at short block lengths, corresponding to a small number of PUF circuits. Open problems in the PUF literature from a signal processing, information theory, coding theory, and hardware complexity perspectives and their combinations are listed to stimulate further advancements in the research on local privacy and security.

Keywords: physical unclonable functions (PUFs); private authentication; secret key generation; information theoretic privacy; code constructions for security

1. Motivations

Fundamental advances in cryptography were made in secret during the 20th century. One exception was Claude E. Shannon's paper "Communication Theory of Secrecy Systems" [1]. Until 1967, the literature on security was not extensive, but a book [2] with a historical review of cryptography changed this trend [3]. Since then, the amount of sensitive data to be protected against attackers has increased significantly. Continuous improvements in security are needed and every improvement creates new possibilities for attacks [4].

Recent hardware-intrinsic security systems, biometric secrecy systems, 5th generation of cellular mobile communication networks (5G) and beyond as well as the emerging internet of things (IoT) networks, have several salient characteristics that differentiate them from existing architectures. These include the deployment of large numbers of possibly low-complexity terminals with light or no infrastructure, stringent constraints on latency, and primary applications of data gathering, inference, and control. These unique characteristics call for a rethinking of some of the fundamentals of data communications and storage. For instance, these characteristics make it very challenging to provide adequate secrecy and privacy primitives. In particular, traditional cryptographic protocols, which require key distribution or certificate management, might not be suitable to support the diverse applications supported by these technologies and might not be able to assure the privacy of personal

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

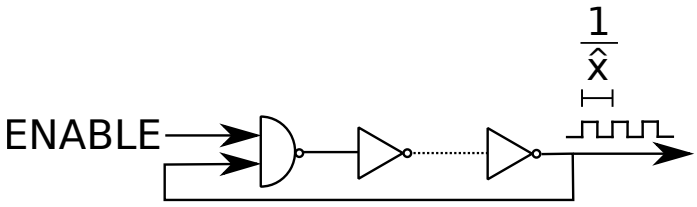

Figure 1. RO logic circuit.

information intrinsic in the data collected by such applications. Furthermore, low-complexity terminals may not have the processing power to implement such protocols, and even if they do, latency tolerances may not permit the processing time needed for cryptographic operations. Similarly, traditional methods of storing a secret key in a secure non-volatile memory (NVM) can be shown to be not secure due to possible invasive attacks to the hardware. Thus, secrecy and privacy for information systems are issues that need to be rethought in the context of recent networks, digital circuits, and database storage.

Information-theoretic security is an emerging approach to provide secrecy and privacy for, e.g., wireless communication systems and networks by exploiting the unique characteristics of the wireless communication channel. Information-theoretic security methods such as physical layer security (PLS) use signal processing, advanced coding, and communication techniques to secure wireless communications at the physical layer. There are two key advantages of PLS. Firstly, it enables the use of resources available at the physical layer such as multiple measurements, channel training mechanisms, power, and rate control, which cannot be utilized by the upper layers of the protocol stack. Secondly, it is based on an information-theoretic foundation for secrecy and privacy that does not make assumptions on the computational capabilities of adversaries, unlike cryptographic primitives. By considering the security and privacy requirements of recent digital systems and the potential benefits from information-theoretic security and privacy methods, it can be seen that information-theoretic methods can complement or even replace conventional cryptographic protocols for wireless networks, databases, and user authentication and identification. Since information-theoretic methods do not generally require pre-shared secret keys, it might considerably simplify the key management in complicated networks. Thus, these methods might be able to fulfill the stringent hardware area constrains of digital devices and latency constraints in certain 5G/6G applications, or to save computations; hence, battery life for low-power devices such as IoT devices. Furthermore, information-theoretic methods offer "built-in" secrecy and privacy, which are generally agnostic to the network infrastructure and provide better scalability as the size of a network or database increases.

A promising local solution to information-theoretic security and privacy problems is a *physical unclonable function (PUF)* [5]. PUFs generate "fingerprints" for physical devices by using their intrinsic and unclonable properties. For instance, consider ring oscillators (ROs) with a logic circuit of multiple inverters serially connected with a feedback of the output of the last inverter into the input of the first inverter, as depicted in Figure 1. RO outputs are oscillation frequencies $1/\hat{x}$, where $\hat{x}$ is the oscillation period, that are unique and uncontrollable since the difference between different RO outputs is caused by submicron random manufacturing variations that cannot be controlled. One can use RO outputs as a source of randomness, called a *PUF circuit*, to extract secret keys that are unique to the digital device that embodies these ROs. The complete method that puts out a unique secret key by using RO outputs is called an *RO PUF*. Similarly, binary static random access memory (SRAM) outputs can be used as a source of randomness to implement SRAM PUFs in almost all digital devices because most digital devices have embedded SRAMs used for data storage. The logic circuit of an SRAM is depicted in Figure 2 and the logically stable states of an SRAM cell are $(\overline{Q}, Q) = (1, 0)$ and $(0, 1)$. During the power-up, the state is undefined if the manufacturer did not fix it. The undefined power-up state of an SRAM cell converges to one of the stable states due to random and uncontrollable mismatch of the inverter parameters, fixed when the SRAM cell is manufactured [6]. There is also random noise in the cell that affects the cell at every power-up. Since the physical mismatch of the cross-coupled inverters

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

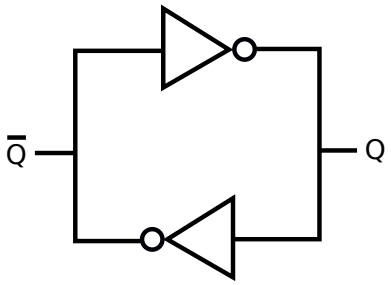

Figure 2. SRAM logic circuit.

is a manufacturing variation, the power-up state of an SRAM cell is considered as a PUF response with one challenge, which is the address of the SRAM cell [6].

PUFs resemble biometric features of human beings. In this review, we will list state-of-the-art methods that bridge the gap between the practical secrecy systems that use PUFs and the information-theoretic security limits by

- Modeling real PUF outputs to solve security problems with valid assumptions;
- Analyzing methods that make information-theoretic analysis tractable, e.g., by transforming PUF symbols so that the transform-domain outputs are almost independent and identically distributed (i.i.d.), and that result in smaller hardware area than benchmark designs in the literature;
- Stating the information-theoretic limits for realistic PUF output models and providing optimal and practical (i.e., low-complexity and finite-length) code constructions that achieve these limits;
- Illustrating best-in-class nested codes for realistic PUF output models.

In short, we start with real PUF outputs to obtain mathematically-tractable models of their behaviour and then list optimal code constructions for these models. Since we discuss methods developed from the fundamentals of signal processing and information theory, any further improvements in this topic are likely to follow the listed steps in this review.

1.1. Organization and Main Insights

This paper is organized as follows. In Section 2, we define a PUF, list its existing and potential applications, and analyze the most promising PUF types. The PUF output models and design challenges faced when manufacturing reliable, low-complexity, and secure PUFs are listed in Section 3. The main security challenge in designing PUFs, i.e., output correlations, is tackled in Section 4 mainly by using a transform coding method, which can provably protect PUFs against various machine learning attacks. The reliability and secrecy performance (e.g., the number of authenticated users) metrics used for PUF designs are defined and jointly optimized in Section 5. PUF security and complexity performance evaluations for the defined transform coding method are given in Section 6. Performance results for error-correction codes used in combination with previous code constructions that are used for key extraction with PUFs, are shown in Section 7 in order to illustrate that previous key extraction methods are strictly suboptimal. We next define the information theoretic metrics and the ultimate key-leakage-storage rate regions for the key agreement with PUFs problem, as well as comparing available code constructions for the key agreement problem in Section 8. Optimal code constructions for the key extraction with PUFs are implemented in Section 9 by using nested polar codes, which are used in 5G networks in the control channel, to illustrate significant gains from using optimal code constructions. In Section 10, we provide a list of open PUF problems that might be interesting for information theorists, coding theorists, and signal processing researchers in addition to the PUF community.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

2. PUF Basics

We give a brief review of the literature on PUFs and discuss the problems with previous PUF designs that can be tackled by using signal processing and coding-theoretic methods.

A PUF is a function that is embodied in a physical device and is unclonable. In the literature, there are alternative expansions of the term PUF such as "physically unclonable function", suggesting that it is a function that is only physically-unclonable. Such PUFs may provide a weaker security guarantee since they allow their functions to be digitally-cloned. For any practical application of a PUF, we need the property of unclonability both physically and digitally. We therefore consider a function as a PUF only if it is a physical function, which is embodied in a physical device, that is unclonable digitally and physically.

Physical identifiers such as PUFs are heuristically defined to be complex challenge-response mappings that depend on the random variations in a physical object. Secret sequences are derived from this complex mapping, which can be used as a secret key. One important feature of PUFs is that the secret sequence generated is not required to be stored and it can be regenerated on demand. This property makes PUFs cheaper (no requirement for a memory for secret storage) and safer (the secret sequence is regenerated only on demand) alternatives to other secret generation and storage techniques such as storing the secret in an NVM [5].

There are an immense number of PUF types, which makes it practically impossible to give a single definition of PUFs that covers all types. We provide the following definition of PUFs that includes all PUF types of interest for this review.

Definition 1 ([5]). *A PUF is a challenge-response mapping embodied by a physical device such that it is fast and easy for the physical device to put out the PUF response and hard for an attacker, who does not have access the PUF circuits, to determine the PUF response to a randomly chosen challenge, even if he has access to a set of challenge-response pairs.*

The terms used in Definition 1, i.e., fast, easy, and hard, are relative terms that should be quantified for each PUF application separately. There are physical functions, called physical one-way functions (POWFs), in the literature that are closely related to PUFs. Such functions are obtained by applying the cryptographic concept of "one-way functions", i.e., functions that are easy to evaluate but (on average) difficult to invert [7], to physical systems. As the first example of POWFs, the speckle pattern obtained from coherent waves propagating through a disordered medium is a one-way function of both the physical randomness in the medium and the angle of the beam used to generate the optical waves [8].

Similar to POWFs, biometric identifiers such as the iris, retina, and fingerprints are closely related to PUFs. Most of the assumptions made for biometric identifiers are satisfied also by PUFs, so we can apply almost all of the results in the literature for biometric identifiers to PUFs. However, it is common practice to assume that PUFs can resist invasive (physical) attacks, which are considered to be the most powerful attacks used to obtain information about a secret in a system, unlike biometric identifiers that are constantly available for attacks. The reason for this assumption is that invasive attacks permanently destroy the fragile PUF outputs [5]. This assumption will be the basis for the PUF system models used throughout this review. We therefore assume that the attacker does not observe a sequence that is correlated with the PUF outputs, unlike biometric identifiers, since physical attacks applied to obtain such a sequence permanently change the PUF outputs.

2.1. Applications of PUFs

A PUF can be seen as a source of random sequences hidden from an attacker who does not have access to the PUF outputs. Therefore, any application that takes a secret sequence as input can theoretically use PUFs. We list some scenarios where PUFs fit well practically:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

- Security of information in wireless networks with an eavesdropper, i.e., a passive attacker, is a PLS problem. Consider Wyner's wiretap channel model introduced in [9], where a transmitter sends a message through a broadcast channel so that a legitimate receiver can reliably reconstruct the message, while the message should be kept secret from an eavesdropper. This model is the most common PLS model, which is a channel coding problem unlike the secret key agrement problem we consider below that is a source coding problem. A randomized encoder helps the transmitter in keeping the message secret by confusing the eavesdropper. Therefore, one can use PUFs at the transmitter as the source of local randomness when a message should be sent securely through the wiretap channel.
- Consider a 5G/6G mobile device that uses a set of SRAM outputs, which are available in mobile devices, as PUF circuits to extract secret keys so that the messages to be sent are encrypted with these secret keys before sending the data over the wireless channel. In this way, the receiver (e.g., a base station) that previously obtained the secret keys can decrypt the data, while an eavesdropper who only overhears the data broadcast over the wireless channel cannot easily learn the message sent.
- The controller area network (CAN) bus standard used in modern vehicles is illustrated in [10] to be susceptible to denial-of-service attacks, which shows that safety-critical inputs of the internal vehicle network such as brakes and throttle can be controlled by an attacker. One countermeasure is to encrypt the transmitted CAN frames by using block ciphers with secret keys generated from PUF outputs used as inputs.
- IoT devices such as wearable or e-health devices may carry sensitive data and use a PUF to store secret keys so that only a mobile device with access to the secret keys can control the IoT devices. One common example of such applications is when PUFs are used to authenticate wireless body sensor network devices [11].
- Cloud storage requires security to protect users' sensitive data. However, securing the cloud is expensive and the users do not necessarily trust the cloud service providers. A PUF in a universal serial bus (USB) token, i.e., Saturnus®, has been trademarked to encrypt user data before uploading the data to the cloud, decrypted locally by reconstructing the same secret from the same PUF.
- System developers want to mutually authenticate a field programmable gate array (FPGA) chip and the intellectual property (IP) components in the chip, and IP developers want to protect the IP. In [12], a protocol is described to achieve these goals with a small hardware area that uses one symmetric cipher and one PUF.

Other applications of PUFs include providing non-repudiation (i.e., undeniable transmission or reception of data), proof of execution on a specific processor, and remote integrated circuit (IC) enabling. Every application of PUFs has different assumptions about the PUF properties, computational complexity, and the specific system models. Therefore, there are different constraints and system parameters for each application. We focus mainly on the application where a secret key is generated from a PUF for user, or device, authentication with privacy and secrecy guarantees, and low complexity.

2.2. Main PUF Types

We review four PUF types, i.e., silicon, arbiter, RO, and SRAM PUFs. We consider mainly the last two PUF types for algorithm and code designs due to their common use in practice and because signal processing techniques can tackle the problems arising in designing these PUFs. For a review of other PUF types that are mostly considered in the hardware design and computer science literatures, and various classifications of PUFs, see, e.g., [4,13,14].

2.3. Silicon and Arbiter PUFs

Common complementary metal-oxide-semiconductor (CMOS) manufacturing processes are used to build silicon PUFs, where the response of the PUF depends on the circuit delays which vary across

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

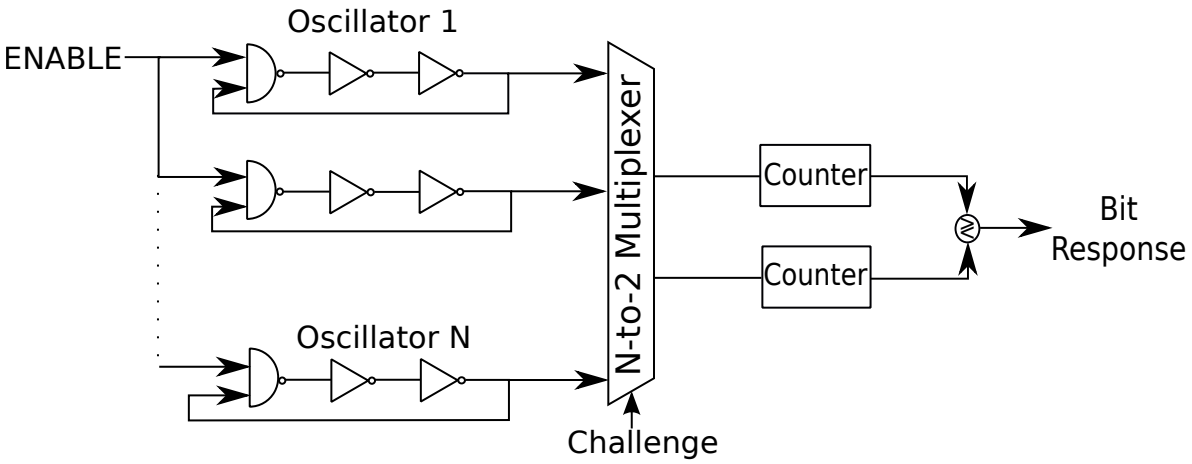

Figure 3. The first and most common RO PUF design [17].

integrated circuits (ICs) [5]. Due to high sensitivity of the circuit delays to environmental changes (e.g., ambient temperature and power supply voltage), arbiter PUFs are proposed in [15], for which an arbiter (i.e., a simple transparent data latch) is added to the silicon PUFs so that the delay comparison result is a single bit. The difference of the path delays is mapped to, e.g., the bit 0 if the first path is faster, and the bit 1 otherwise. The difference between the delays can be small, causing meta-stable outputs. Since the output of the mapper is generally preset to 0, the incoming signals must satisfy the setup time (t_{setup}) of the latch to switch the output to 1, resulting in a bias in the arbiter PUF outputs. Symmetrically implementable latches (e.g., set-reset latches) should be used to overcome this problem, which is difficult because FPGA routing does not allow the user to enforce symmetry in the hardware implementation. We discuss below that PUFs without symmetry requirements, e.g., RO PUFs, provide better results.

2.4. Ring Oscillator PUFs

The logic circuit of an odd number of inverters serially connected, where the output of the last inverter is given as input to the first inverter is an RO, as depicted in Figure 1. The first logic gate in Figure 1 is a NAND gate, giving the same logic output as an inverter gate when the ENABLE signal is 1 (ON), to enable/disable the RO circuit. The manufacturing-dependent and uncontrollable component in an RO is the total propagation delay of an input signal to flow through the RO, determining the oscillation frequency $1/\hat{x}$ of an RO that is used as the source of randomness. A self-sustained oscillation is possible when the ring provides a 2π phase shift and has unit voltage gain at the oscillation frequency $1/\hat{x}$.

Consider an RO with $m \geq 3$ inverters. Each inverter should provide a phase shift of $\frac{\pi}{m}$ with an additional phase shift of π due to the feedback. Therefore, the signal should flow through the RO twice to provide the necessary phase shift [16]. Suppose a propagation delay of τ_d for each inverter, so the oscillation frequency of an RO is $\frac{1}{\hat{x}} = \frac{1}{2m\tau_d}$. We remark that since RO outputs are generally measured by using, e.g., 32-bit counters, it is realistic to assume that measured RO outputs are realizations of a continuous distribution, as assumed below.

The propagation delay τ_d is affected by nonlinearities in the digital circuit. Furthermore, there are deterministic noise sources such as the cross-talk between adjacent signal traces and additional random noise sources such as thermal noise and flicker noise [16]. Such effects should be eliminated to have a reliable RO output. Rather than improving the standard RO designs, which would impose the condition that manufacturers should change their RO designs, the first proposal to fix the reliability problem was to make hard bit decisions by comparing RO pairs [17], as illustrated in Figure 3.

In Figure 3, the multiplexers are challenged by a bit sequence of length at most $\lceil \log_2 N \rceil$ so that an RO pair is selected among N ROs. The counters put out the number of rising edges from each

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

RO for a fixed time duration. A logic bit decision is made by comparing the counter values, which can be bijectively mapped to the oscillation frequencies. For instance, when the upper RO has a greater counter value, then the bit 0 is generated; otherwise, the bit 1. Given that ROs are identically laid out in the hardware, the differences in the oscillation frequencies are determined mainly by uncontrollable manufacturing variations. Furthermore, it is not necessary to have a symmetric layout when hard-macro hardware designs are used for different ROs, unlike arbiter PUFs.

The key extraction method illustrated in Figure 3 gives an output of $\binom{N}{2}$ bits, which are correlated due to overlapping RO comparisons. This causes a security threat and makes the RO PUF vulnerable to various attacks, including machine learning attacks. Thus, non-overlapping pairs of ROs are used in [17] to extract each bit. However, there are systematic variations in the neighboring ROs due to the surrounding logic, which also should be eliminated to extract sequences with full entropy. Furthermore, ambient temperature and supply voltage variations are the most important effects that reduce the reliability of RO PUF outputs. A scheme called *1-out-of-k masking* is proposed as a countermeasure to these effects, which compares the RO pairs that have the maximum oscillation frequency differences for a range of voltages and temperatures to extract bits [17] . The bits extracted by such a comparison are more reliable than the bits extracted by using previous methods. The main disadvantages of this scheme are that it is inefficient due to unused RO pairs, and only a single bit is extracted from the (semi-) continuous RO outputs. We review transform-coding based RO PUF methods below that significantly improve on these methods.

2.5. SRAM PUFs

There are multiple memory-based PUFs such as SRAM, Flip-flop, DRAM, and Butterfly PUFs. Their common feature is to posses a small number of challenge-response pairs with respect to their sizes. As the most promising memory-based PUF type that is already used in the industry, we consider SRAM PUFs that use the uncontrollable settling state of bi-stable circuits [18]. In the standard SRAM design, there are four transistors used to form the logic of two cross-coupled inverters, as depicted in Figure 2, and two other transistors to access the inverters. The power-up state, i.e., $(\overline{Q}, Q) = (1, 0)$ or $(0, 1)$, of an SRAM cell provides one secret bit. Concatenating many such bits allows to generate a secret key from SRAM PUFs on demand. We provide an open problem about SRAM PUFs below.

3. Correlated, Biased, and Noisy PUF Outputs

PUF circuit outputs are biased (nonuniform), correlated (dependent), and noisy (erroneous). We review a transform-coding algorithm that extracts an almost i.i.d. uniform bit sequence from each PUF, so a helper-data generation algorithm can correct the bit errors in the sequence generated from noisy PUF outputs. Using this transform-coding algorithm, we also obtain memoryless PUF measurement-channel models, so standard information-theoretic tools, which cannot be easily applied to correlated sequences, can be used.

Remark 1. *The bias in the PUF circuit outputs is considered in the PUF literature to be a big threat against the security of the key generated from PUFs since the bias allows to apply, e.g., machine learning attacks. However, it is illustrated in [19, Figure 6] that the output bias does not change the information-theoretic rate regions significantly, illustrating that there exist code constructions that do not require PUF outputs to be uniformly distributed.*

There are multiple *key-generation*, i.e., *generated-secret (GS)*, and *key-binding*, i.e., *chosen-secret (CS)*, methods to reconstruct secret keys from noisy PUF outputs, where the key is generated from the PUF outputs or bound to them, respectively. A code-offset fuzzy extractor (COFE) [20] is an example of key-generation methods and the fuzzy-commitment scheme (FCS) [21] is a key-binding method. Since a secret key should be stored in a secure database for both models, it might be practical to allow a trusted entity to choose the secret key that is bound to a PUF output. Thus, we first analyse a method

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

that significantly improves reliability, privacy, secrecy, and hardware cost performance by using a transform-coding algorithm that is applied to PUF outputs in combination with the FCS. We remark that the information-theoretic analysis of the CS model follows directly from the analysis of the GS model [22], so one can use either model for comparisons.

Correlation in PUF outputs might leak information about the secret key, called *secrecy leakage*, and about the PUF output, called *privacy leakage* [22,23]. Moreover, noise reduces the reliability of PUF outputs and error-correction codes are needed to satisfy the stringent reliability constraints. The transform-coding approach proposed in [24] in combination with a set of scalar quantizers has made its way into secret-key binding with continuous-output identifiers. This approach allows to significantly reduce the output correlation and to adjust the effective noise at the PUF output with reliability guarantees.

3.1. PUF Output Model

Consider a (semi-)continuous output physical function such as an RO output as a source that puts out a real-valued symbol $\hat{x}$. Systematic variations in RO outputs in a two-dimensional array are less than the systematic variations in one-dimensional ROs, since in a two-dimensional array the maximum distance between RO hardware logic circuits is less, which decreases the variations in the RO outputs caused by surrounding hardware logic circuits [25]. Thus, consider a two-dimensional RO array of size $l = r \times c$ and represent the array as a vector random variable $\widehat{X}^l$. Suppose there is a single two-dimensional RO array in each device with the same circuit design and the RO array emits an output $\widehat{X}^l$ according to a probability density $f_{\widehat{X}^l}$. Each RO output is disturbed by mutually-independent additive Gaussian noise and the vector noise is denoted as $\widehat{Z}^l$. Define the noisy RO outputs as $\widehat{Y}^l = \widehat{X}^l + \widehat{Z}^l$. Observe that $\widehat{X}^l$ and $\widehat{Y}^l$ are correlated. A secret key can thus be agreed by using these outputs [26,27].

One needs to extract random sequences with i.i.d. symbols from $\widehat{X}^l$ and $\widehat{Y}^l$ to employ available information-theoretic results in [28] for secret-key binding with identifiers by using the FCS. An algorithm is proposed in [24] that extracts almost i.i.d. binary and uniformly distributed random vectors X^n and Y^n from $\widehat{X}^l$ and $\widehat{Y}^l$, respectively. For such X^n and Y^n, we can define a binary error vector as $E^n = X^n \oplus Y^n$, where $\oplus$ is the modulo-2 sum. The random sequence E^n corresponds to a sequence of i.i.d. Bernoulli random variables with parameter p, i.e., $E^n \sim \text{Bern}^n(p)$. The channel $P_{Y|X}$ is thus a binary symmetric channel (BSC) with crossover probability p, i.e., BSC(p). We discuss a transform-coding method below, which further provides reliability guarantees for each bit generated.

The FCS can reconstruct a secret key by using correlated random variables without leaking any information about the secret key [21]. The FCS is depicted in Figure 4, where an encoder $\text{Enc}(\cdot)$ maps a secret key $S \in \mathcal{S}$, which is uniformly distributed in the set $\{1, 2, \ldots, |\mathcal{S}|\}$, into a binary codeword C^n that is added modulo-2 to the binary PUF-output sequence X^n during enrollment. The resulting sequence is called helper data W, sent through a public and noiseless communication link to a database. The modulo-2 sum of the helper data W and Y^n gives the result $R^n = W \oplus Y^n = C^n \oplus E^n$, which can be later mapped to an estimate $\hat{S}$ of the secret key S by the decoder $\text{Dec}(\cdot)$ during reconstruction.

We next give information-theoretic rate regions for the FCS; see [29] for information-theoretic notation and basics.

Definition 2. *A secret-key vs. privacy-leakage rate pair (R_s, R_ℓ) is achievable by the FCS with perfect secrecy, i.e., zero secrecy leakage, if, given any $\epsilon > 0$, there is some $n \geq 1$, and an encoder and decoder for which $R_s = \dfrac{\log |\mathcal{S}|}{n}$ and*

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

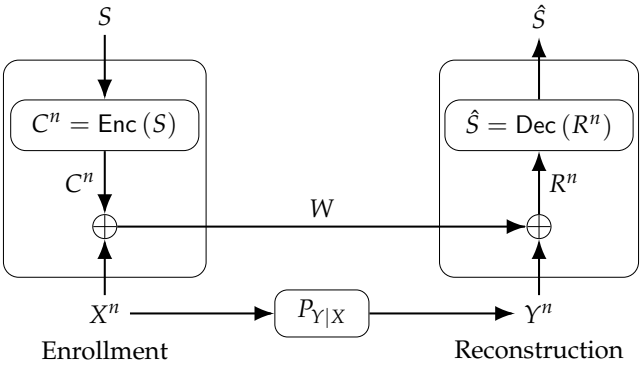

Figure 4. The fuzzy commitment scheme (FCS).

$$P_B = \Pr[S \neq \hat{S}] \leq \epsilon \qquad (\textit{reliability}) \tag{1}$$
$$H(S) \geq n(R_s - \epsilon) \qquad (\textit{key uniformity}) \tag{2}$$
$$I(S;W) = 0 \qquad (\textit{perfect secrecy}) \tag{3}$$
$$I(X^n;W) \leq n(R_\ell + \epsilon) \qquad (\textit{privacy}) \tag{4}$$

where (3) suggests that S and W are independent and (4) suggests that the rate of dependency between X^n and W is bounded. The achievable secret-key vs. privacy-leakage rate, or key-leakage, region $\mathcal{R}_{FCS}$ for the FCS is the union of all achievable pairs.

Theorem 1 ([28]). *The key-leakage region $\mathcal{R}_{FCS}$ for the FCS with a channel $P_{Y|X}$ that is a BSC(p), uniformly distributed X and Y, and zero secrecy leakage is*

$$\mathcal{R}_{FCS} = \{(R_s, R_\ell): \quad 0 \leq R_s \leq 1 - H_b(p), \qquad R_\ell \geq 1 - R_s\} \tag{5}$$

where $H_b(p) = -p\log p - (1-p)\log(1-p)$ is the binary entropy function.

The region $\mathcal{R}_{\text{FCS}}$ suggests that any (secret-key, privacy-leakage) rate pair that sums to 1 bit/source-bit is achievable with the constraint that the secret-key rate is at most the channel capacity of the BSC(p), i.e., $\max_{p_X} I(X;Y) = 1 - H_b(p)$. Furthermore, smaller secret-key rates and greater privacy-leakage rates than these rates are also achievable.

The FCS is a particular realization of the CS model. The region $\mathcal{R}$ of all achievable (secret-key, privacy-leakage) rate pairs for the CS model with a negligible secrecy-leakage rate, where a generic encoder is used to confidentially transmit an embedded secret key to a decoder that observes Y^n and the helper data W, is given in [22] as

$$\mathcal{R} = \bigcup_{P_{U|X}} \left\{ (R_s, R_\ell): \quad 0 \leq R_s \leq I(U;Y), \qquad R_\ell \geq I(U;X) - I(U;Y) \right\} \tag{6}$$

where $U - X - Y$ forms a Markov chain and the alphabet $\mathcal{U}$ of the auxiliary random variable U can be limited to have the size $|\mathcal{U}| \leq |\mathcal{X}| + 1$. The auxiliary random variable U represents a distorted version of X through a channel $P_{U|X}$. The FCS is optimal, i.e., it achieves a boundary point of $\mathcal{R}$, for a BSC $P_{Y|X}$ with crossover probability p only at the point $(R_s^*, R_\ell^*) = (1 - H_b(p), H_b(p))$ [28]. This point corresponds to the highest achievable secret-key rate; see Figure 7 below. We remark that the region $\mathcal{R}$ gives an outer bound for the perfect-secrecy case considered to obtain $\mathcal{R}_{\text{FCS}}$.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

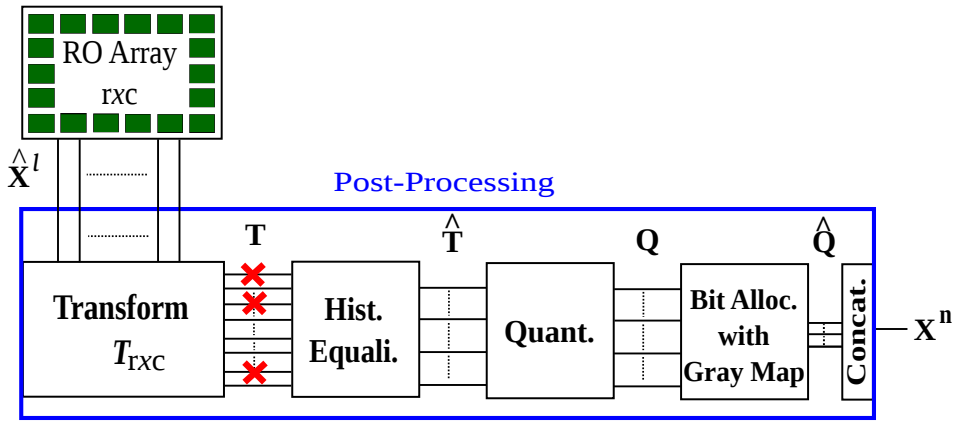

Figure 5. Transform-coding steps [24].

4. Transform Coding Steps

The main aim of transform coding is to reduce correlations between outputs of the ROs in the same two-dimensional array by using, e.g., a linear transformation. We discuss a transform-coding algorithm proposed in [30] as an extension of [24] to provide reliability guarantees to each generated bit. Joint optimization of the quantizer and error-correction code parameters to maximize the security and reliability performance, and a simple method to decrease hardware storage are its main steps. The output of these post-processing steps is a bit sequence X^n (or its noisy version Y^n) used in the FCS. One applies the same post-processing steps for the enrollment and reconstruction. The difference is that during enrollment the design parameters are chosen as a function of the PUF-circuit output statistics by the device manufacturer. It thus suffices to discuss only the enrollment steps. Figure 5 shows the post-processing steps that include transformation, histogram equalization, quantization, bit allocation, and bit-sequence concatenation.

RO outputs $\widehat{X}^l$ are correlated due to, e.g., the surrounding logic in the hardware. A transform $T_{r\times c}(\cdot)$ of size $r \times c$ is applied to an array of RO outputs to reduce correlations. Decorrelation performance of a transform depends on the source statistics. We model each real-valued output T in the transform domain, called *transform coefficient*, obtained from an RO-output dataset in [31] by using the corrected Akaike's information criterion (AICc) [32] and the Bayesian information criterion (BIC) [33]. These criteria suggest that a Gaussian distribution can be fitted to each transform coefficient T for the DCT, discrete Walsh-Hadamard transform (DWHT), discrete Haar transform (DHT), and Karhunen-Loève transform (KLT), which are common orthogonal transforms considered in the literature for image processing, digital watermarking, etc. Use maximum-likelihood estimation to derive unbiased estimates for the parameters of Gaussian distributions.

The histogram equalization step in Figure 5 converts the probability density of the i-th coefficient T_i into a standard normal distribution such that $\widehat{T}_i = \frac{T_i - \mu_i}{\sigma_i}$, where μ_i is the mean and σ_i is the standard deviation of the i-th transform coefficient for all $i = 1, 2, \ldots, l$. Quantization steps for all transform coefficients are thus the same. Without histogram equalization, we need a different quantizer for each transform coefficient. Therefore, the histogram equalization step reduces the storage for the quantization steps. Transformed and equalized coefficients $\widehat{T}_i$ are independent if the transform $T_{r\times c}(\cdot)$ decorrelates the RO outputs perfectly and the transform coefficients T_i are jointly Gaussian. One can thus use a scalar quantizer for all coefficients without a performance loss for this case. Scalar quantizers and bit extraction methods are given below to satisfy the security and reliability requirements of the FCS with the independence assumption, which can be combined with a correlation-thresholding approach in practice.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

5. Joint Quantizer and Error-Correction Code Design

The aim of the post-processing steps in Figure 5 is to extract a uniformly-random bit sequence X^n. Use a quantizer $\Delta(\cdot)$ with quantization-interval values $k = 1, 2, \cdots, 2^{K_i}$, where K_i is the number of bits we extract from the i-th coefficient $\widehat{T}_i$ for $i = 1, 2, \ldots, l$. Let

$$\Delta(\hat{t}_i) = k \quad \text{if} \quad b_{k-1} < \hat{t}_i \leq b_k \tag{7}$$

and choose $b_k = \Phi^{-1}\left(\frac{k}{2^{K_i}}\right)$, where $\Phi^{-1}(\cdot)$ is the quantile function of the standard normal distribution. The output k is assigned to a bit sequence of length K_i. The most likely error event when we quantize $\widehat{T}_i$ is a jump to a neighboring quantization step due to zero-mean noise model assumed. Thus, apply a Gray mapping when assigning bit sequences of length K_i to the integers $k = 1, 2, \ldots, 2^{K_i}$, so neighboring bit sequences change only in one bit.

We next discuss a reliability metric proposed for a joint quantizer and code design by fixing the maximum number of erroneous transform coefficients and considering an error-correction code that can correct all error patterns with up to a fixed number of errors.

5.1. Quantizer Design with Fixed Maximum Number of Errors

We discuss a conservative approach that assumes that either all bits extracted from a transform coefficient are correct or they all flip. The *correctness probability* P_c of a transform coefficient is defined to be the probability that all bits associated with this coefficient are correct. This metric is used to determine the number of bits extracted from each coefficient such that there is a channel encoder and a bounded minimum distance decoder (BMDD) that satisfy the block-error probability constraint $P_B \leq 10^{-9}$, which is a common block-error probability considered for PUFs that consist of CMOS circuits [17]. This approach results in reliability guarantees for the random-output RO arrays.

Let $Q(\cdot)$ be the Q-function, $\sigma^2_{\hat{n}}$ the noise variance, and $f_{\widehat{T}}$ the probability density of the standard Gaussian distribution. For a K-bit quantizer and the quantization boundaries b_k as in (7) for an equalized Gaussian transform coefficient $\widehat{T}$, the correctness probability is

$$P_c(K) = \sum_{k=0}^{2^K-1} \int_{b_k}^{b_{k+1}} \left[Q\left(\frac{b_k - \hat{t}}{\sigma_{\hat{n}}}\right) - Q\left(\frac{b_{k+1} - \hat{t}}{\sigma_{\hat{n}}}\right) \right] f_{\widehat{T}}(\hat{t}) d\hat{t}. \tag{8}$$

Suppose the channel decoder can correct all errors in up to $C_{\max}$ transform coefficients. Suppose further that coefficient errors occur independently, i.e., noise on different transform coefficients are mutually independent, and that the correctness probability $P_{c,i}(K)$ of the i-th coefficient $\widehat{T}_i$ for $i = 1, 2, \ldots, l$ is at least $\overline{P}_c(C_{\max})$. A sufficient condition for satisfying the block-error probability constraint $P_B \leq 10^{-9}$ is that $\overline{P}_c(C_{\max})$ satisfies the inequality

$$\sum_{c=C_{\max}+1}^{l} \binom{l}{c} (1 - \overline{P}_c(C_{\max}))^c \overline{P}_c(C_{\max})^{l-c} \leq 10^{-9}. \tag{9}$$

Determine the number K_i of bits extracted from the i-th transform coefficient as the maximum value K such that $P_{c,i}(K) \geq \overline{P}_c(C_{\max})$. The first coefficient, i.e., DC coefficient, $\widehat{T}_1$ is not used since its value is a scaled version of the mean of the RO outputs in the same array, which is generally known by an attacker. Ambient-temperature and supply-voltage variations have a highly-linear effect on the RO outputs, so the DC coefficient is the most affected coefficient, which is another reason not to use the DC coefficient [34]. Therefore, choose $K_1 = 0$ so that the total number $n(C_{\max})$ of extracted bits is

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

$$n(C_{\max}) = \sum_{i=2}^{l} K_i. \tag{10}$$

In the worst case, the coefficients in error are the coefficients from which the largest number of bits are extracted. Sort the numbers K_i of bits extracted from all coefficients in descending order such that $K'_i \geq K'_{i+1}$ for all $i = 1, 2, \ldots, l-2$. The channel decoder thus must be able to correct up to

$$e(C_{\max}) = \sum_{i=1}^{C_{\max}} K'_i \tag{11}$$

bit errors, which can be satisfied by using a block code with minimum distance $d_{\min} \geq 2e(C_{\max})+1$ since a BMDD can correct all errors with up to $e = \lfloor \frac{d_{\min}-1}{2} \rfloor$ errors [35].

Suppose a key bound to physical identifiers in a device is used in the advanced encryption standard (AES) with a uniformly-distributed secret key of length 128 bits. The block code used in the FCS should thus have a code length of at most $n(C_{\max})$ bits, code dimension of at least 128 bits, and minimum distance of $d_{\min} \geq 2e(C_{\max}) + 1$ for a fixed $C_{\max}$. The code rate should be as high as possible to operate close to the optimal (secret-key, privacy-leakage) rate point of the FCS. This optimization problem is hard to solve. One can illustrate by an exhaustive search over a set of $C_{\max}$ values and over a selection of algebraic codes that there is a channel code that satisfies these constraints with a reliability guarantee for each extracted bit. Restricting the search to codes that admit low-complexity encoders and decoders is desired for, e.g., IoT applications, for which complexity is the bottleneck.

Conditions listed above are conservative. For a given transform coefficient, the correctness probability can be significantly greater than the correctness threshold $\overline{P_c}(C_{\max})$. Secondly, due to Gray mapping, it is more likely that less than K_i bits are in error when the i-th coefficient is erroneous. It is also unlikely that the bit errors always occur in the transform coefficients from which the largest numbers of bits are extracted. Thus, even if a channel code cannot correct all error patterns with up to $e(C_{\max})$ errors, it can still be the case that the block-error probability constraint is satisfied. We next illustrate such a case.

6. PUF Performance Evaluations

Suppose RO outputs $\widehat{X}^l$ are represented as a vector random variable with an autocovariance matrix $\mathbf{C}_{\widehat{\mathbf{X}}\widehat{\mathbf{X}}}$. Consider RO arrays of sizes 8×8 and 16×16. Autocovariance matrix of such RO array outputs and noise parameters can be estimated from the RO output dataset in [31]. Using the dataset, we compare the performance of the DCT, DWHT, DHT, and KLT in terms of their decorrelation efficiency, complexity, uniqueness, and security.

6.1. Decorrelation Performance

One should eliminate correlations between the RO outputs and make them independent to extract uniform bit sequences by quantizing each transform coefficient separately. Use the *decorrelation efficiency* η_c [36] as a decorrelation performance metric. Consider the autocovariance matrix $\mathbf{C}_{\mathbf{TT}}$ of the transform coefficients, so η_c of a transform is

$$\eta_c = 1 - \frac{\sum_{a=1}^{l}\sum_{b=1}^{l} |\mathbf{C}_{\mathbf{TT}}(a,b)| \mathbb{1}\{a \neq b\}}{\sum_{a=1}^{l}\sum_{b=1}^{l} |\mathbf{C}_{\widehat{\mathbf{X}}\widehat{\mathbf{X}}}(a,b)| \mathbb{1}\{a \neq b\}} \tag{12}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

where the indicator function $\mathbb{1}\{a \neq b\}$ takes on the value 1 if $a \neq b$ and 0 otherwise. The decorrelation efficiency of the KLT is 1, which is optimal [36]. We list the average decorrelation efficiency results of other transforms in Table 1. All transforms have similar and good decorrelation efficiency performance for the RO outputs in the dataset [31]. The DCT and DHT have the highest efficiency for 8×8 RO arrays, whereas for 16×16 RO arrays, the best transform is the DWHT. Table 1 indicates that increasing the array size improves η_c.

Table 1. The average RO output decorrelation-efficiency results.

	DCT	DWHT	DHT
η_c for 8×8	0.9978	0.9977	0.9978
η_c for 16×16	0.9987	0.9988	0.9986

6.2. Transform Complexity

We measure the complexity of a transform in terms of the number of operations required to compute the transform. We are interested in a computational-complexity comparison for RO arrays of sizes $r = c = 8$ and $r = c = 16$, which are powers of 2, so that fast algorithms are available for the DCT, DWHT, and DHT. The computational complexity of the KLT for $r = c = n$ is $O(n^3)$, while it is $O(n^2 \log_2 n)$ for the DCT and DWHT, and $O(n^2)$ for the DHT [37]. There are efficient implementations of the DWHT without multiplications [30], which can be applied also to the transforms proposed in [38]. The DWHT is thus a good candidate for RO PUF designs for IoT applications.

6.3. Uniqueness and Security

The bit sequence extracted from a physical identifier should be uniformly distributed to make the rate region $\mathcal{R}_{\text{FCS}}$ in (5) valid. A common measure, called *uniqueness*, for measuring randomness of bit sequences is the average fractional Hamming distance between the bit sequences extracted from different RO PUFs. Similar uniqueness results are obtained for all transforms, where the mean Hamming distance is 0.500 and Hamming distance variance is approximately 7×10^{-4}. All transforms thus provide close to optimal uniqueness results due to their high decorrelation efficiencies and equipartitioned quantization intervals, which are better than previous RO PUF results with mean values of 0.462 [17] and 0.473 [31].

The national institute of standards and technology (NIST) provides a set of randomness tests that check whether a bit sequence can be differentiated from a uniformly random bit sequence [39]. Apply these tests to measure the randomness of the generated sequences. The bit sequences generated from ROs in the dataset [31] with the DWHT pass most of the applicable tests for short lengths, which is considered to be an acceptable result [39]. The KLT performs the best due to its optimal decorrelation performance. One can apply a thresholding approach such that the reliable transform coefficients from which the bits are extracted do not have high correlations, which further improves the security performance.

7. Error-Correction Codes for PUFs with Transform Coding

Suppose that bit sequences extracted by using the transform-coding method are i.i.d. and uniformly distributed so that the secrecy leakage is zero. We assume that signal processing steps mentioned above perform well, so we can conduct standard information- and coding-theoretic analysis. We list different codes designed for the transform-coding algorithm according to the reliability metric considered above.

Select a channel code for the quantizer designed above for a fixed maximum number of errors to store a secret key of length 128 bits. The correctness probabilities defined in (8) for the transform coefficients with the three highest and three smallest probabilities are plotted in Figure 6. The indices of the 16×16 transform coefficients follow the order in the dataset [31], where the coefficient index at

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

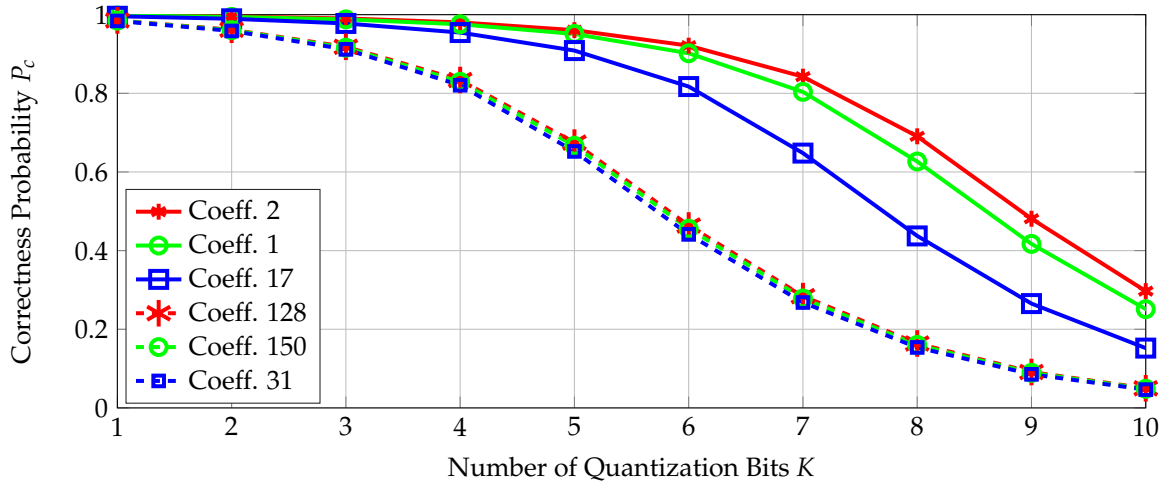

Figure 6. The correctness probabilities for transform coefficients.

the first row and first column is 1, and it increases columnwise up to 16 so that the second row starts with the index 17, the third row with the index 33, etc. The most reliable transform coefficients are the low-frequency coefficients, which are in our case at the upper-left corner of the two-dimensional (2D) transform-coefficient array with indices such as $1, 2, 3, 17, 18, 19, 33, 34$, and 35. The low-frequency transform coefficients therefore have the highest signal-to-noise ratios (SNRs) for the source and noise statistics obtained from the RO dataset [31]. The least reliable coefficients are observed to be spatially away from the transform coefficients at the upper-left or lower-right corners of the 2D transform-coefficient array. These results indicate that the *SNR-packing efficiency*, which can be defined similarly as the energy-packing efficiency, of a transform follows a more complicated scan order than the classic zig-zag scan order used for the energy-packing efficiency. Observe from Figure 6 that increasing the number of extracted bits decreases the correctness probability for all coefficients since the quantization boundaries get closer so that errors due to noise become more likely, i.e., the probability $P_c(K)$ defined in (8) decreases with increasing K.

Fix the maximum number $C_{\max}$ of transform coefficients allowed to be in error and calculate the correctness threshold $\overline{P}_c(C_{\max})$ using (9), the total number $n(C_{\max})$ of extracted bits using (10), and the number $e(C_{\max})$ of errors the block code should be able to correct using (11). Observe that if $C_{\max} \leq 10$, $\overline{P}_c(C_{\max})$ is so large that $P_{c,i}(K=1) \leq \overline{P}_c(C_{\max})$ for all $i = 2, \ldots, l$. If $11 \leq C_{\max} \leq 15$, $n(C_{\max})$ is less than the required code dimension of 128 bits. Furthermore, increasing $C_{\max}$ results in a smaller correctness threshold $\overline{P}_c(C_{\max})$ so that the maximum of the number $K_{\max}(C_{\max}) = K'_1(C_{\max})$ of bits extracted among the $l-1$ used coefficients increases. This result can increase hardware complexity. Therefore, consider only the cases where $C_{\max} \leq 20$. Table 2 shows $\overline{P}_c(C_{\max})$, $n(C_{\max})$, and $e(C_{\max})$ for a range of $C_{\max}$ values used for channel-code selection.

Table 2. Code-parameter constraints.

$C_{\max}$	16	17	18	19	20
$\overline{P}_c$	0.9902	0.9889	0.9875	0.9860	0.9844
$K_{\max}$	3	3	3	3	3
n	144	224	250	255	259
e	18	20	21	23	25

Consider binary (extended) Bose–Chaudhuri–Hocquenghem (BCH) and Reed-Solomon (RS) codes, which have good minimum-distance $d_{\min}$ properties. An exhaustive search does not provide a code with dimension of at least 128 bits and with parameters satisfying any of the $(n(C_{\max}), e(C_{\max}))$ pairs in Table 2. However, the correctness threshold analysis leading to Table 2 is conservative. Thus, choose a BCH code with parameters as close as possible to an $(n(C_{\max}), e(C_{\max}))$ pair, for which one

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

can prove that even if the number e_{BCH} of errors the chosen BCH code can correct is less than $e(C_{\max})$, the block-error probability constraint is satisfied. Consider the binary BCH code with the block length 255, code dimension 131, and a capability of correcting all error patterns with up to $e_{\mathrm{BCH}} = 18$ errors.

First, impose the condition that exactly one bit is extracted from each coefficient, i.e., $K_i = 1$ for all $i = 2, 3, \ldots, l$, so that in total $n = l - 1 = 255$ bits are obtained, which results in mutually independent bit errors E_i. Therefore, the chosen block code should be able to correct all error patterns with up to $e = 20$ bit errors rather than $e(20) = 25$ bit errors, which is still greater than the error-correction capability $e_{\mathrm{BCH}} = 18$ of the considered BCH code.

The block error probability P_B for the BCH code $\mathcal{C}(255, 131, 37)$ with a BMDD corresponds to the probability of having more than 18 errors in the codeword, i.e.,

$$P_B = \sum_{j=19}^{255} \left[\sum_{A \in \mathcal{F}_j} \prod_{i \in A} (1 - P_{c,i}) \cdot \prod_{i \in A^c} P_{c,i} \right] \tag{13}$$

where $P_{c,i}$ is the correctness probability of the i-th transform coefficient $\widehat{T}_i$ defined in (8) for $i = 2, 3, \ldots, 256$, $\mathcal{F}_j$ is the set of all size-j subsets of the set $\{2, 3, \ldots, 256\}$, and A^c denotes the complement of the set A. The correctness probabilities $P_{c,i}$ are different and they represent probabilities of independent events due to the independence assumption for the transform coefficients. We use the discrete Fourier transform characteristic function method [40] to calculate the block-error probability and obtain the result $P_B \approx 1.26 \times 10^{-11} < 10^{-9}$. The block-error probability constraint is thus satisfied by using the BCH code $\mathcal{C}(255, 131, 37)$ with a BMDD although the conservative analysis suggests that it would not be satisfied.

We next compare the BCH code $\mathcal{C}(255, 131, 37)$ with previous codes proposed for binding keys to physical identifiers with the FCS and a secret-key length of 128 bits such that $P_B \leq 10^{-9}$ is satisfied. The (secret-key, privacy-leakage) rate pair for this code is $(R_{\mathrm{s}}, R_\ell) = (\frac{131}{255}, 1 - \frac{131}{255}) \approx (0.514, 0.486)$ bits/source-bit. This pair is significantly better than previous results. The main reason for obtaining a better (secret-key, privacy-leakage) rate pair is that the quantizer defined above allows to obtain a higher identifier-output reliability by decreasing the number of bits extracted from a transform coefficient.

Compare the secret-key R_{s} and privacy-leakage R_ℓ rates of the BCH code $\mathcal{C}(255, 131, 37)$ with the region of all achievable rate pairs for the CS model and the FCS for a BSC $P_{Y|X}$ with crossover probability $p_b = 1 - \frac{1}{l-1} \sum_{i=2}^{l} P_{c,i}(K_i = 1) \approx 0.0097$, i.e., the probability of being in error averaged over all used transform coefficients with the quantizer defined above. Compute the boundary points of the region $\mathcal{R}$ by finding the optimal auxiliary random variable U in (6) when $P_{Y|X}$ is a BSC. The regions of all rate pairs achievable by the FCS and CS model, the maximum secret-key rate point, the (secret-key, privacy-leakage) rate pair of the BCH code, and a finite-length (non-asymptotic) bound [41] for the block length of $n = 255$ bits and $P_B = 10^{-9}$ are plotted in Figure 7.

The maximum secret-key rate is $R_{\mathrm{s}}^* \approx 0.922$ bits/source-bit with a corresponding minimum privacy-leakage rate of $R_\ell^* \approx 0.079$ bits/source-bit. There is a gap between the rate tuple achieved by the BCH code and the only operation point where the FCS is optimal, i.e., $(R_\ell^*, R_{\mathrm{s}}^*)$. Part of this rate loss can be explained by the short block length of the code and the small block-error probability constraint. The finite-length bound given in [41, Theorem 52] establishes that the rate pair $(R_{\mathrm{s}}, R_\ell) = (0.691, 0.309)$ bits/source-bit is achievable by using the FCS, as depicted in Figure 7. One can therefore further improve the rate pairs by using better channel codes and decoders with higher hardware complexity, but this may not be possible for IoT applications. Figure 7 also illustrates that there exist other code constructions (other than standard channel codes) that reduce the privacy-leakage rate as well as the storage rate for each fixed secret-key rate, which we consider below.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

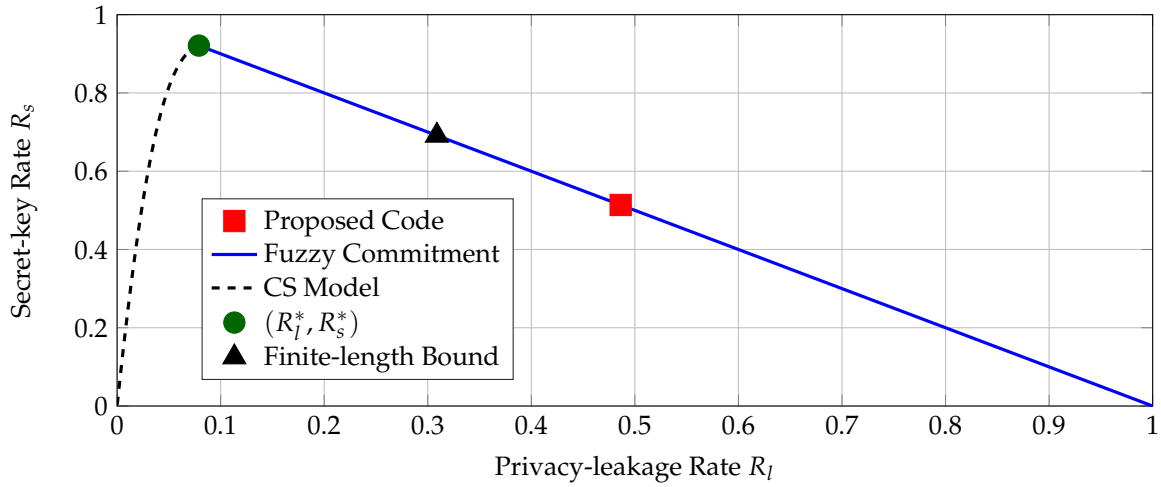

Figure 7. The operation point of the considered BCH code $\mathcal{C}(255, 131, 37)$, regions of achievable rate pairs according to (5) and (6), the maximum secret-key rate point (R_ℓ^*, R_s^*), and a finite-length bound for $n = 255$ bits, $P_B = 10^{-9}$, and BSC (0.0097).

8. Code Constructions for PUFs

Consider the two-terminal key agreement problem, where the identifier outputs during enrollment are noiseless. We mention two optimal linear code constructions from [42] that are based on Wyner-Ziv (WZ) coding [43]. The first construction uses random linear codes and achieves all points of the key-leakage-storage regions of the GS and CS models. The second construction uses nested polar codes for vector quantization during enrollment and for error correction during reconstruction. Simulations show that nested polar codes achieve privacy-leakage and storage rates that improve on existing code designs, and one designed code achieves a rate tuple that cannot be achieved by existing methods.

Several practical code constructions for key agreement with identifiers have been proposed in the literature. For instance, the COFE and the FCS both require a standard error-correction code to satisfy the constraints of, respectively, the key generation (GS model) and key embedding (CS model) problems, as discussed above. Similarly, a polar code construction is proposed in [44] for the GS model. These constructions are shown to be suboptimal in terms of the privacy-leakage and storage rates.

The binary Golay code is used in [22] as a vector quantizer (VQ) in combination with Slepian-Wolf (SW) codes [45] to illustrate that the key vs. storage (or key vs. leakage) rate ratio can be increased via quantization. This observation motivates the use of a VQ to improve the performance of previous constructions. We next consider VQ by using WZ coding to decrease storage rates. The WZ-coding construction turns out to be optimal, which is not coincidental. For instance, the bounds on the storage rate of the GS model and on the WZ rate (storage rate) have the same mutual information terms optimized over the same conditional probability distribution. This similarity suggests an *equivalence* that is closely related to the concept of *formula duality*. In fact, the optimal random code construction, encoding, and decoding operations are identical for both problems. One therefore can call the GS model and WZ problem *functionally equivalent*. Such a strong connection suggests that there might exist constructive methods that are optimal for both problems for all channels, which is closely related to the *operational duality* concept.

Consider the GS model in Figure 8(a), where a secret key is generated from a biometric or physical source. During enrollment, the encoder observes an i.i.d. noiseless sequence X^n, generated by the source according to some P_X, and computes a secret key S and public helper data W as $(S, W) = \mathsf{Enc}(X^n)$. During reconstruction, the decoder observes a noisy source measurement Y^n of the source X^n through a memoryless channel $P_{Y|X}$ together with the helper data W. The decoder estimates the secret key as $\widehat{S} = \mathsf{Dec}(Y^n, W)$. Similarly, Figure 8(b) shows the CS model, where a secret key S that is independent of (X^n, Y^n) is embedded into the helper data as $W = \mathsf{Enc}(X^n, S)$. The decoder for

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

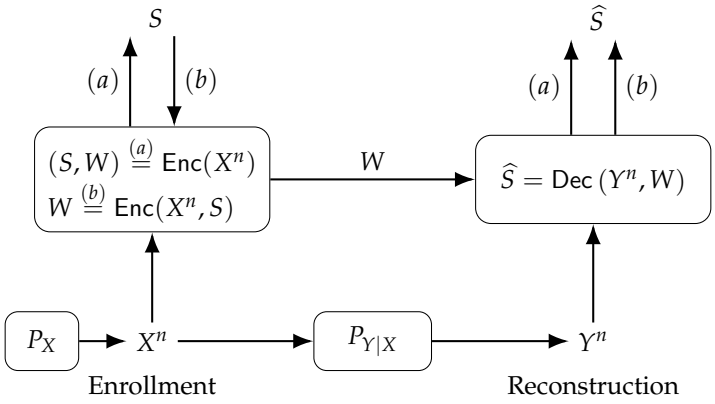

Figure 8. The (a) GS and (b) CS models.

the CS model estimates the secret key as $\widehat{S} = \mathsf{Dec}(Y^n, W)$. The source, measurement, secret key, and storage alphabets $\mathcal{X}$, $\mathcal{Y}$, $\mathcal{S}$, and $\mathcal{W}$ are finite sets, which can be achieved if, e.g., the transform-coding algorithm discussed above is applied.

Definition 3. *A key-leakage-storage tuple* (R_s, R_ℓ, R_w) *is* achievable *for GS and CS models if, given any* $\epsilon > 0$*, there is some* $n \geq 1$*, an encoder, and a decoder such that* $R_s = \dfrac{\log|\mathcal{S}|}{n}$ *and*

$$P_B = \Pr[S \neq \hat{S}] \leq \delta \qquad (reliability) \tag{14}$$
$$I(S;W) \leq n\delta \qquad (weak\ secrecy) \tag{15}$$
$$I(X^n;W) \leq n(R_\ell + \delta) \qquad (privacy) \tag{16}$$
$$H(S) \geq n(R_s - \delta) \qquad (uniformity) \tag{17}$$
$$\log|\mathcal{W}| \leq n(R_w + \delta) \qquad (storage) \tag{18}$$

are satisfied. The key-leakage-storage *regions* $\mathcal{R}_{gs}$ *and* $\mathcal{R}_{cs}$ *for the GS and CS models, respectively, are the closures of the sets of achievable tuples for the corresponding models.*

Theorem 2 ([22])**.** *The key-leakage-storage regions for the GS and CS models, respectively, are*

$$\mathcal{R}_{gs} = \bigcup_{P_{U|X}} \Big\{ (R_s, R_\ell, R_w) : \begin{array}{l} 0 \leq R_s \leq I(U;Y), \\ R_\ell \geq I(U;X) - I(U;Y), \\ R_w \geq I(U;X) - I(U;Y) \end{array} \Big\},$$

and

$$\mathcal{R}_{cs} = \bigcup_{P_{U|X}} \Big\{ (R_s, R_\ell, R_w) : \begin{array}{l} 0 \leq R_s \leq I(U;Y), \\ R_\ell \geq I(U;X) - I(U;Y), \\ R_w \geq I(U;X) \end{array} \Big\}$$

where $U - X - Y$ *form a Markov chain. These regions are convex sets. The alphabet* $\mathcal{U}$ *of the auxiliary random variable* U *can be limited to have size* $|\mathcal{U}| \leq |\mathcal{X}| + 1$ *for both regions.*

Remark 2. *One can improve the weak secrecy to strong secrecy, i.e., we can replace (15) with* $I(S;W) \leq \epsilon$ *by applying information reconciliation and privacy amplification steps to multiple blocks of identifier outputs as described in [46], e.g., by using multiple PUFs in a device for key agreement.*

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

Assume, as above, that $X^n \sim \text{Bern}^n(\frac{1}{2})$ and the channel $P_{Y|X}$ is a BSC(p_A), where $p_A \in [0, 0.5]$. Define the star-operation as $q * p_A = q(1-p_A) + (1-q)p_A$. The key-leakage-storage region of the GS model is

$$\begin{aligned}\mathcal{R}_{\text{gs,bin}} = \bigcup_{q\in[0,0.5]} \Big\{ & (R_{\text{s}}, R_\ell, R_{\text{w}}) : \\ & 0 \le R_{\text{s}} \le 1 - H_b(q * p_A), \\ & R_\ell \ge H_b(q * p_A) - H_b(q), \\ & R_{\text{w}} \ge H_b(q * p_A) - H_b(q) \Big\}. \end{aligned} \tag{19}$$

8.1. Comparisons Between Code Constructions for PUFs

There are several existing code constructions proposed for the GS and CS models. Consider the three best methods: FCS for the CS model, and COFE and the polar code construction in [44] for the GS model, to compare them with the WZ-coding constructions. Similar steps to the FCS are applied in the COFE, except that the secret key is a hashed version of X^n. The FCS achieves the single optimal point in the key-leakage region with the maximum secret-key rate $R_{\text{s}}^* = I(X;Y)$; the privacy-leakage rate is $R_\ell^* = H(X|Y)$. Similarly, the COFE achieves the same boundary point in the key-leakage region. This is, however, the only boundary point of the key-leakage regions that these methods can achieve.

One can improve both methods by adding a VQ step: instead of X^n we use its quantized version X_q^n during enrollment. This asymptotically corresponds to summing the original helper data and an independent random variable $J^n \sim \text{Bern}^n(q)$ such that $W = X^n \oplus C^n \oplus J^n$ is the new helper data so that we create a virtual channel $P_{Y|X\oplus J}$ and apply the FCS or COFE to this virtual channel. The modified FCS and COFE can achieve all points of the key-leakage region if we take a union of all rate pairs achieved over all $q \in [0, 0.5]$. However, the helper data have n bits for both methods, and the resulting storage rate of 1 bit/source-bit is not necessarily optimal.

The polar code construction in [44] requires less storage rate than the FCS and COFE. However, this approach improves only the storage rate and cannot achieve all points of the key-leakage-storage region. Furthermore, in [44] some code designs assume that there is a "private" key shared only between the encoder and decoder, which is not realistic since a private key requires hardware protection against invasive attacks. If such a protection is possible, then there is no need to use an on-demand key reconstruction method like a PUF.

The existing methods cannot, therefore, achieve all points of the key-leakage-storage region for a BSC, unlike the WZ-coding constructions described in [42] and illustrated with nested polar code designs below. In previous works, only the secret-key rates of the proposed codes are compared because the sum of the secret-key and privacy-leakage rates is one. This constraint means that increasing the key vs. leakage (or key vs. storage) rate ratio is equivalent to increasing the key rate. Instead, WZ-coding constructions are more flexible than the existing methods in terms of achievable rate tuples. Therefore, use the key vs. storage rate ratio as the metric to control the storage and privacy leakage.

9. Optimal Code Constructions with Polar Codes

Polar codes [47] have a low encoding/decoding complexity, asymptotic optimality for various information-theoretic problems, and good finite length performance if a list decoder is used in combination with an outer code. Furthermore, they have a structure that allows a simple nested code design and they can be used for WZ coding [48].

Polar codes rely on the *channel polarization* phenomenon, where a channel is converted into polarized bit channels by a polar transform. This transform converts an input sequence U^n with frozen

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

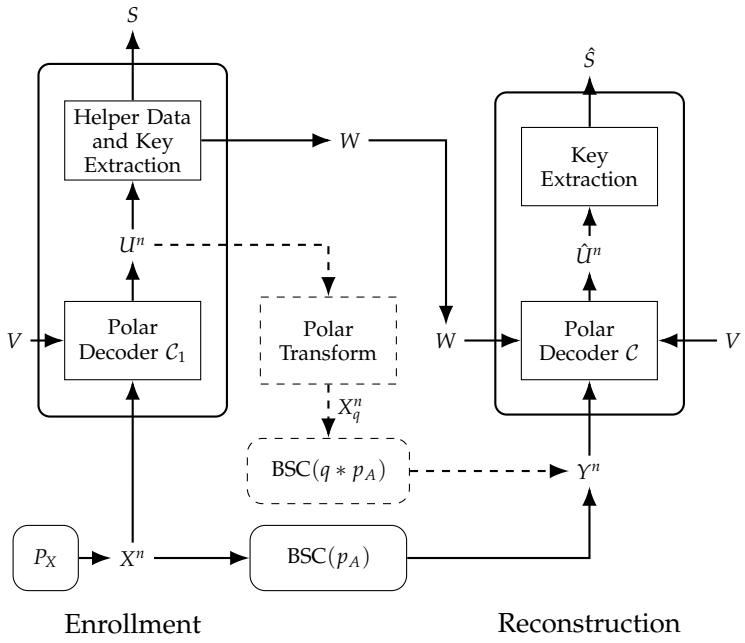

Figure 9. Second WZ-coding construction for the GS model.

and unfrozen bits to a codeword of the same length n. A polar decoder processes a noisy observation of the codeword together with the frozen bits to estimate U^n.

Let $\mathcal{C}(n, \mathcal{F}, G^{|\mathcal{F}|})$ denote a polar code of block length n, where $\mathcal{F}$ is the set of indices of the frozen bits and $G^{|\mathcal{F}|}$ is the sequence of frozen bits. In the following, we use the nested polar code construction proposed in [48] for WZ coding.

9.1. Polar Code Construction for the GS Model

Consider two polar codes $\mathcal{C}_1(n, \mathcal{F}_1, V)$ and $\mathcal{C}(n, \mathcal{F}, \overline{V})$ with $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_w$ and $\overline{V} = [V, W]$, where V has length m_1 and W has length m_2 such that m_1 and m_2 satisfy

$$\frac{m_1}{n} = H_b(q) - \delta \qquad \text{and} \qquad \frac{m_1 + m_2}{n} = H_b(q * p_A) + \delta \tag{20}$$

for some distortion $q \in [0, 0.5]$ and $\delta > 0$. The indices in $\mathcal{F}_1$ represent frozen channels with assigned values V for both codes and $\mathcal{C}$ has additional frozen channels with assigned values W denoted by the set of indices $\mathcal{F}_w$, so the codes are nested.

The code $\mathcal{C}_1$ serves as a VQ with a desired distortion q since its rate is greater than the lossy source coding capacity with average distortion q. The code $\mathcal{C}$ serves as the error-correction code for a BSC$(q * p_A)$ since its rate is less than channel capacity of this channel. The idea is to obtain W during enrollment and store it as public helper data. For reconstruction, (W, V, Y^n) are used by the decoder to estimate the secret key S of length $n - m_1 - m_2$. Figure 9 shows the block diagram of this construction. Suppose V is the all-zero vector so that no additional storage is necessary. This choice has no effect on the average distortion $E[q]$ between X^n and X_q^n defined below; see [48, Lemma 10].

Enrollment: The uniform binary sequence X^n generated by a PUF during enrollment is treated as the noisy observation of a BSC(q). X^n is quantized by a polar decoder of $\mathcal{C}_1$. Extract from the decoder output U^n the bits at indices $\mathcal{F}_w$ and store them as the helper data W. The bits at the indices $j \in \{1, 2, \ldots, n\} \setminus \mathcal{F}$ are used as the secret key. Note that applying the polar transform to U^n generates X_q^n, which is a distorted version of X^n. The distortion between X^n and X_q^n is modeled as a BSC(q) because the error sequence $E_q^n = X^n \oplus X_q^n$ resembles an i.i.d. sequence $\sim \text{Bern}^n(q)$ when $n \to \infty$ [48, Lemma 11].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

Reconstruction: During reconstruction, the polar decoder of $\mathcal{C}$ observes the binary sequence Y^n, which is a noisy measurement of X^n through a BSC(p_A). The frozen bits $\overline{V} = [V, W]$ at indices $\mathcal{F}$ are given to the polar decoder. The output $\widehat{U}^n$ of the polar decoder is the estimate of U^n and contains the estimate $\widehat{S}$ of the secret key at the unfrozen indices of $\mathcal{C}$, i.e., $j \in \{1, 2, \ldots, n\} \setminus \mathcal{F}$.

We next summarise a method to design practical nested polar codes for the GS model.

Construction of $\mathcal{C}$ and $\mathcal{C}_1$: Since $\mathcal{C} \subseteq \mathcal{C}_1$ are nested codes, they must be constructed jointly. $\mathcal{F}$ and $\mathcal{F}_1$ should be chosen such that the reliability and security constraints are satisfied. For a given secret key size $n - m_1 - m_2$, block length n, crossover probability p_A, and target block-error probability $P_B = \Pr[S \neq \widehat{S}]$, consider the following nested polar code design procedure [42].

1. Construct a polar code of rate $(n-m_1-m_2)/n$ and use it as the code $\mathcal{C}$, i.e., define the set of frozen indices $\mathcal{F}$.
2. Evaluate the error correction performance of $\mathcal{C}$ with a decoder for a BSC over a range of crossover probabilities to obtain the crossover probability p_c, resulting in a target block-error probability of P_B. Using $p_c = E[q] * p_A$, we obtain the target distortion $E[q] = (p_c - p_A)/(1 - 2p_A)$ averaged over a large number of realizations of X^n.
3. Find an $\mathcal{F}_1 \subset \mathcal{F}$ that results in an average distortion of $E[q]$ with a minimum possible amount of helper data. Use $\mathcal{F}_1$ as the frozen set of $\mathcal{C}_1$.

Step 1 is a conventional channel code design task, similar to the codes designed for the transform-coding algorithm above, and step 2 is applied by Monte-Carlo simulations. For step 3, we start with $\mathcal{F}_1' = \mathcal{F}$ and compute the resulting average distortion $E[q']$ via Monte-Carlo simulations. If $E[q']$ is not less than $E[q]$, remove elements from $\mathcal{F}_1'$ according to the reliabilities of the polarized bit channels and repeat the procedure until we obtain the desired average distortion $E[q]$.

The distortion level introduced by the VQ is an additional degree of freedom in choosing the code design parameters. For instance, different values of P_B can be targeted with the same code by changing the distortion level. Alternatively, devices with different p_A values can be supported by using the same code. This additional degree of freedom makes the mentioned code design suitable for a wide range of applications.

9.2. Designed Polar Codes for the GS Model

Consider, e.g., the GS model where S is used in the AES and $\log|\mathcal{S}| = n - m_1 - m_2 = 128$ bits, as considered above. If we use PUFs in an FPGA as the randomness source, we must satisfy a block-error probability P_B of at most 10^{-6}. Consider a BSC $P_{Y|X}$ with crossover probability $p_A = 0.15$, which is a common value for SRAM PUFs under ideal environmental conditions [6] and for RO PUFs under varying environmental conditions [34]. Consider nested polar codes for these parameters to illustrate that one can achieve better key-leakage-storage rate tuples than previously proposed codes.

Code 1: Consider $n = 1024$ bits and recall that $n - m_1 - m_2 = 128$ bits, $P_B = 10^{-6}$, and $p_A = 0.15$. Polar successive cancellation list (SCL) decoders with list size 8 are used as the VQ and channel decoder. First, design the code $\mathcal{C}$ of rate $128/1024$ and evaluate its performance with the SCL decoder for a BSC with a range of crossover probabilities. A block-error probability of 10^{-6} is observed at a crossover probability of $p_c = 0.1819$. Since $p_A = 0.15$, this corresponds to an average distortion of $E[q] = 0.0456$ and the target average distortion is obtained at $n - m_1 = 778$ bits. Thus, $m_2 = 650$ bits of helper data suffice to obtain a block-error probability of $P_B = 10^{-6}$.

Code 2: Consider the same parameters as in Code 1, except $n = 2048$ bits. Apply the same steps as above. A crossover probability of $p_c = 0.2682$ is required to obtain a block-error probability of 10^{-6}, which gives an average distortion of $E[q] = 0.1689$. The target average distortion is achieved with helper data of length 611 bits.

The error probability P_B is calculated as an average over a large number of PUF realizations, i.e., over a large number of PUF devices with the same circuit design. To satisfy the block-error probability requirement for each PUF realization, one could consider using the maximum distortion instead of

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

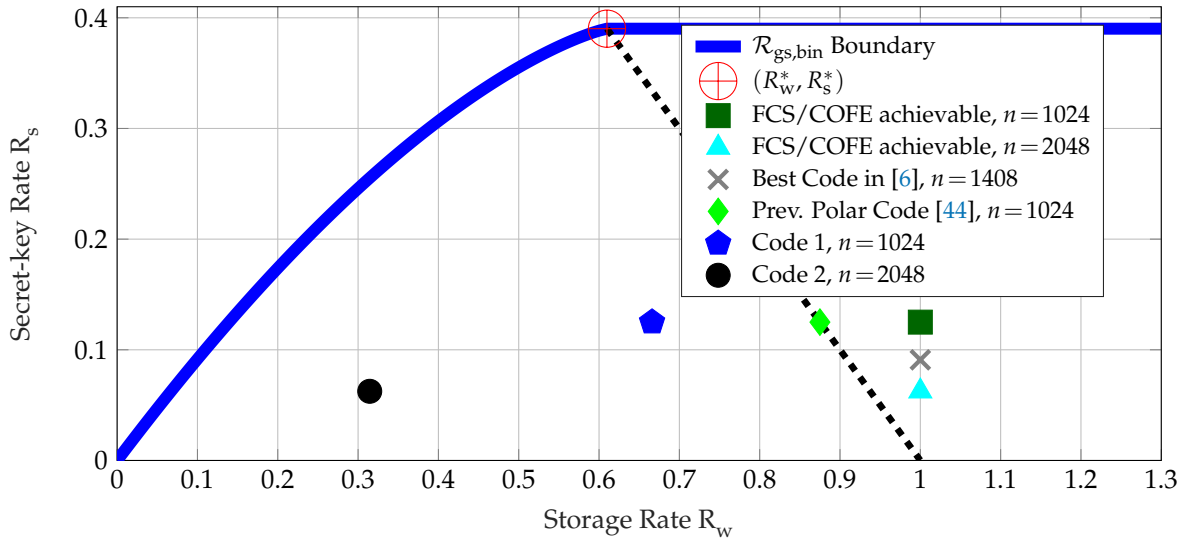

Figure 10. Storage-key rates for the GS model with $p_A = 0.15$. The (R_w^*, R_s^*) point is the best possible point achieved by SW-coding constructions, which lies on the dashed line representing $R_w + R_s = H(X)$. The block error probability satisfies $P_B \leq 10^{-6}$ and the key length is 128 bits for all code points.

$E[q]$ in step 3 of the design procedure given above. This would increase the amount of helper data. One can guarantee for the considered parameters a block-error probability of at most 10^{-6} for 99.99% of all realizations x^n of X^n by adding 32 bits to the helper data for Code 1 and 33 bits for Code 2. The numbers of extra helper data bits required are small since the variance of the distortion q over all PUF realizations is small for the blocklengths considered. For comparisons, we consider the helper data sizes required to guarantee $P_B = 10^{-6}$ for 99.99% of all PUF realizations.

9.3. Code Comparisons

Figure 10 depicts the storage-key (R_w, R_s) projection of the boundary points of the region $\mathcal{R}_{gs,bin}$ for $p_A = 0.15$. Furthermore, we show the point with the maximum secret-key rate R_s^* and the minimum storage rate R_w^* to achieve R_s^*. For the FCS and COFE, use the random coding union bound [41, Thm. 16] to confirm that the plotted rate pairs are achievable for a secret-key length of 128 bits, an error probability of $P_B = 10^{-6}$, and blocklengths of $n = 1024$ and $n = 2048$. These rate pairs are shown in Figure 10 to the right of the dashed line representing $R_w + R_s = 1$ bit/source-bit. Similarly, the rate pairs achieved by the previous polar code design in [44], and Codes 1 and 2 are shown in Figure 10.

Storage rates of the FCS and COFE are 1 bit/source-bit, which is suboptimal. The previous polar code construction in [44] achieves a rate point with $R_s + R_w = 1$ bit/source-bit, which is expected since this is a SW-coding construction. The previous polar code construction improves on the rate pairs achieved by the FCS and COFE in terms of the key vs. storage ratio. Nested polar codes achieve the key-leakage-storage rates of approximately $(0.125, 0.666, 0.666)$ bits/source-bit by Code 1 and $(0.063, 0.315, 0.315)$ bits/source-bit by Code 2, projections of which are depicted in Figure 10. These rates are significantly better than all previous code constructions for the same parameters and without any private key assumption. Designed nested polar codes increase the ratio R_s/R_w from approximately 0.188 for Code 1 to 0.199 for Code 2, suggesting to increase the blocklength to obtain better ratios. Code 2 achieves privacy-leakage and storage rates that cannot be achieved by previous methods without applying the method of *time sharing*, since Code 2 achieves privacy-leakage and storage rates of 0.315 bits/source-bit that are significantly less than the minimum privacy-leakage and storage rates $R_w^* = R_\ell^* = H_b(p_A) \approx 0.610$ bits/source-bit that can be asymptotically achieved by previous methods.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 November 2020 doi:10.20944/preprints202011.0209.v1

Apply the sphere packing bound [49, Eq. (5.8.19)] to upper bound the key vs. storage rate ratio that can be achieved by SW-coding constructions for the maximum secret-key rate point. Consider $p_A = 0.15$, $n = 1024$, and $P_B = 10^{-6}$, for which the sphere packing bound requires that the rate of the code $\mathcal{C}$ satisfies $R_{\mathcal{C}} \leq 0.273$. Assume that the key rate is given by its maximal value $R_{\mathrm{s}} = R_{\mathcal{C}}$ and the storage rate is given by its minimal value $R_{\mathrm{w}} = 1 - R_{\mathcal{C}}$, then we arrive at $R_{\mathrm{s}}/R_{\mathrm{w}} \leq 0.375$. A similar calculation for $n = 2048$ yields $R_{\mathrm{s}}/R_{\mathrm{w}} \leq 0.437$. These results indicate that there are still gaps between the maximum key vs. storage rate ratios achieved by WZ-coding constructions and the ratios achieved by Codes 1 and 2. The gaps can be reduced by using different nested polar codes that improve the minimum-distance properties, as in [50], or by using nested algebraic codes for which design methods are available in the literature, as in [51]. We remark again that such optimality-seeking approaches, e.g., based on information-theoretic security, provide the right insights into the best solutions for the digital era's security and privacy problems.

10. Discussions and Open Problems

- We want to use low-complexity scalar quantizers after transformation without extra secrecy leakage; however, the decorrelation efficiency metric does not fully represent the dependency between transform coefficients. What is the right metric to use for choosing the transform used in combination with scalar quantizers? Is mutual information between transform coefficients an appropriate metric for this purpose? The choice of the transform should also depend on a reliability metric such as SNR-packing efficiency so that the transform, quantizers, and the error-correction codes can be designed jointly. What is the right reliability metric for this purpose?
- It is shown in [34] that the ambient temperature and supply voltage affect the RO outputs deterministically rather than adding extra random noise, which was assumed in the RO PUF literature. What are the right output models for common PUF types, i.e., what are the deterministic and random components, and how are they related?
- SRAM PUFs are already used in products. In the literature there is no extensive analysis of the output correlations between different SRAMs in the same device possibly because SRAM outputs are binary and it is difficult to model the correlation between binary symbols. However, SRAM outputs are modeled in [6] as binary-quantized sums of independent Gaussian random variables. Is it possible to determine or approximate the correlations between the Gaussian random variables of different SRAMs? If yes, this might be useful for an attacker to obtain information about the secret sequence generated from the SRAM PUF output, which causes extra secrecy leakage.
- The transform-coding approach discussed above results in reliability guarantees for the random-output RO arrays, which considers an average over all ROs manufactured. The worst case scenario is when the transform coefficient value is on the quantization boundary, for which the secret-key capacity is 0 bit. If one replaces the average reliability metric used above by a lower bound on the reliability of each RO, i.e., a worst-case scenario metric, how would this change the rate of the error-correction code used? For a fixed code, what should be the optimal bound on the reliability of each RO to maximize the yield, i.e., the percentage of ROs among all manufactured ROs for which the worst-case reliability guarantee is satisfied?
- Are the WZ problem and the GS model *operationally equivalent*?
- Linear block-code constructions discussed above are for uniformly-distributed PUF outputs. Can one construct other (random) linear block codes that are asymptotically optimal for non-uniform PUF outputs? Is it necessary to use an extensions of the COFE for this purpose?
- Consider the nested polar code design procedure given above. It is not possible to construct a code with this procedure for $n \leq 512$ since $q * p_A$ is an increasing function of q for any $q \in [0, 0.5]$. Is a nested polar code construction possible for $n = 512$ if one improves the code design procedure and the decoder?

Author Contributions: O. Günlü conceived the study, designed, and performed the experiments; Both authors analyzed the data and contributed to the writing of the paper, and a joint effort improved the algorithms presented.

Funding: Authors were supported by the German Federal Ministry of Education and Research (BMBF) within the national initiative for "Post Shannon Communication (NewCom)" under the Grant 16KIS1004.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Shannon, C.E. Communication theory of secrecy systems. *Bell Labs Tech. J.* **1949**, *28*, 656–715.
2. Kahn, D. *The Codebreakers: The Story of Secret Writing*; New York, NY: Macmillan Publishers, 1967.
3. Schneier, B. *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2 ed.; John Wiley & Sons, 1996.
4. Böhm, C.; Hofer, M. *Physical Unclonable Functions in Theory and Practice*; New York, NY: Springer-Verlag, 2012.
5. Gassend, B.; Clarke, D.; Dijk, M.V.; Devadas, S. Silicon physical random functions. ACM Conf. Computer Commun. Security; , 2002; pp. 148–160.
6. Maes, R.; Tuyls, P.; Verbauwhede, I. A Soft Decision Helper Data Algorithm for SRAM PUFs. IEEE Int. Symp. Inf. Theory; 2009; pp. 2101–2105.
7. Goldreich, O. *Modern Cryptography, Probabilistic Proofs and Pseudorandomness*; Vol. 17, Berlin Heidelberg, Germany: Springer-Verlag, 1998.
8. Pappu, R. Physical one-way functions. PhD thesis, M.I.T., Cambridge, MA, 2001.
9. Wyner, A.D. The wire-tap channel. *The Bell Sys. Techn. J.* **1975**, *54*, 1355–1387.
10. Palanca, A.; Evenchick, E.; Maggi, F.; Zanero, S. A stealth, selective, link-layer denial-of-service attack against automotive networks. Int. Conf. Detection Intrusions Malware Vulnerability Assessment; 2017; pp. 185–206.
11. Lee, Y.S.; Lee, H.J.; Alasaarela, E. Mutual authentication in wireless body sensor networks (WBSN) based on Physical Unclonable Function (PUF). Int. Wireless Commun. Mobile Comput. Conf.; 2013; pp. 1314–1318.
12. Simpson, E.; Schaumont, P. Offline hardware/software authentication for reconfigurable platforms. Int. Workshop Cryp. Hardware Embedded Sys.; 2006; pp. 311–323.
13. Herder, C.; Yu, M.; Koushanfar, F.; Devadas, S. Physical Unclonable Functions and Applications: A Tutorial. *IEEE* **2014**, *102*, 1126–1141.
14. Huth, C.; Guillaume, R.; Strohm, T.; Duplys, P.; Samuel, I.A.; Güneysu, T. Information reconciliation schemes in physical-layer security: A survey. *Elsevier Comput. Netw.* **2016**, *109*, 84–104.
15. Lim, D.; Lee, J.W.; Gassend, B.; Suh, G.E.; Dijk, M.V.; Devadas, S. Extracting Secret Keys From Integrated Circuits. *IEEE Trans. Very Large Scale Integration Sys.* **2005**, *13*, 1200–1205.
16. Mandal, M.K.; Sarkar, B.C. Ring oscillators: Characteristics and Applications. *Indian J. Pure Appl. Physics* **2010**, *48*, 136–145.
17. Suh, G.E.; Devadas, S. Physical Unclonable Functions for Device Authentication and Secret Key Generation. ACM Des. Automation Conf.; 2007; pp. 9–14.
18. Guajardo, J.; Kumar, S.S.; Schrijen, G.J.; Tuyls, P. FPGA Intrinsic PUFs and Their Use for IP Protection. Int. Workshop Cryp. Hardware Embedded Syst.; 2007; pp. 63–80.
19. Günlü, O.; Kramer, G.; Skorski, M. Privacy and secrecy with multiple measurements of physical and biometric identifiers. IEEE Conf. Commun. Network Security; 2015; pp. 89–94.
20. Dodis, Y.; Ostrovsky, R.; Reyzin, L.; Smith, A. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.* **2008**, *38*, 97–139.
21. Juels, A.; Wattenberg, M. A fuzzy commitment scheme. ACM Conf. Comp. Commun. Security; 1999; pp. 28–36.
22. Ignatenko, T.; Willems, F.M.J. Biometric systems: Privacy and secrecy aspects. *IEEE Trans. Inf. Forensics Security* **2009**, *4*, 956–973.
23. Günlü, O.; Kramer, G. Privacy, Secrecy, and Storage With Multiple Noisy Measurements of Identifiers. *IEEE Trans. Inf. Forensics Security* **2018**, *13*, 2872–2883.

24. Günlü, O.; İşcan, O. DCT Based Ring Oscillator Physical Unclonable Functions. IEEE Int. Conf. Acoustics Speech Sign. Process.; 2014; pp. 8198–8201.
25. Maiti, A.; Schaumont, P. Improved ring oscillator PUF: an FPGA-friendly secure primitive. *J. Cryptology* **2011**, *24*, 375–397.
26. Ahlswede, R.; Csiszár, I. Common Randomness in Information Theory and Cryptography - Part I: Secret Sharing. *IEEE Trans. Inf. Theory* **1993**, *39*, 1121–1132.
27. Maurer, U.M. Secret Key Agreement by Public Discussion from Common Information. *IEEE Trans. Inf. Theory* **1993**, *39*, 733–742.
28. Ignatenko, T.; Willems, F.M. Information leakage in fuzzy commitment schemes. *IEEE Trans. Inf. Forensics Security* **2010**, *5*, 2337–2348.
29. Cover, T.M.; Thomas, J.A. *Elements of Information Theory*; Hoboken, NJ: John Wiley & Sons, 2012.
30. Günlü, O.; Kernetzky, T.; İşcan, O.; Sidorenko, V.; Kramer, G.; Schaefer, R.F. Secure and Reliable Key Agreement with Physical Unclonable Functions. *Entropy* **2018**, *20*.
31. Maiti, A.; others. A Large Scale Characterization of RO-PUF. IEEE Int. Symp. Hardware-Orient. Security Trust; 2010; pp. 94–99.
32. Sugiura, N. Further analysis of the data by Akaike's information criterion and the finite corrections. *Commun. Stat.-Theory Methods* **1978**, *7*, 13–26.
33. Schwarz, G. Estimating the dimension of a model. *Annals Stat.* **1978**, *6*, 461–464.
34. Günlü, O.; İşcan, O.; Kramer, G. Reliable secret key generation from physical unclonable functions under varying environmental conditions. IEEE Int. Workshop Inf. Forensics Security; 2015; pp. 1–6.
35. Lin, S.; Costello, D.J. *Error Control Coding*; Englewood Cliffs, NJ: Prentice-Hall, 2004.
36. Ohm, J.R. *Multimedia Signal Coding and Transmission*; Berlin Heidelberg, Germany: Springer-Verlag, 2015.
37. Wang, R. *Introduction to Orthogonal Transforms: With Applications in Data Processing and Analysis*; Cambridge, U.K.: Cambridge Univ. Press, 2012.
38. Günlü, O.; Schaefer, R.F. Low-Complexity and Reliable Transforms for Physical Unclonable Functions. IEEE Int. Conf. Acoustics, Speech Sign. Process.; 2020; pp. 2807–2811.
39. Rukhin, A.; others. A statistical test suite for random and pseudorandom number generators for cryptographic applications. Technical report, National Inst. Stand. Techno., 2001. Rev. in 2010.
40. Hong, Y. On computing the distribution function for the sum of independent and nonidentical random indicators. Technical report, Dep. Stat., Virginia Tech., Blacksburg, VA, 2011.
41. Polyanskiy, Y.; Poor, H.V.; Verdú, S. Channel Coding Rate in the Finite Blocklength Regime. *IEEE Trans. Inf. Theory* **2010**, *56*, 2307–2359.
42. Günlü, O.; İşcan, O.; Sidorenko, V.; Kramer, G. Code Constructions for Physical Unclonable Functions and Biometric Secrecy Systems. *IEEE Trans. Inf. Forensics Security* **2019**, *14*, 2848–2858.
43. Wyner, A.D.; Ziv, J. The rate-distortion function for source coding with side information at the decoder. *IEEE Trans. Inf. Theory* **1976**, *22*, 1–10.
44. Chen, B.; Ignatenko, T.; Willems, F.M.; Maes, R.; van der Sluis, E.; Selimis, G. A Robust SRAM-PUF Key Generation Scheme Based on Polar Codes. IEEE Global Commun. Conf.; 2017; pp. 1–6.
45. Slepian, D.; Wolf, J. Noiseless coding of correlated information sources. *IEEE Trans. Inf. Theory* **1973**, *19*, 471–480.
46. Maurer, U.; Wolf, S. Information-theoretic key agreement: From weak to strong secrecy for free. Int. Conf. Theory Appl. Cryptographic Techn.; 2000; pp. 351–368.
47. Arikan, E. Channel Polarization: A Method for Constructing Capacity-Achieving Codes for Symmetric Binary-Input Memoryless Channels. *IEEE Trans. Inf. Theory* **2009**, *55*, 3051–3073.
48. Korada, S.B.; Urbanke, R.L. Polar Codes are Optimal for Lossy Source Coding. *IEEE Trans. Inf. Theory* **2010**, *56*, 1751–1768.
49. Gallager, R.G. *Low-Density Parity-Check Codes*; Cambridge, MA: M.I.T. Press, 1963.
50. Günlü, O.; Trifonov, P.; Kim, M.; Schaefer, R.F.; Sidorenko, V. Randomized Nested Polar Subcode Constructions for Privacy, Secrecy, and Storage. to appear in IEEE Int. Symp. Inf. Theory Appl.; 2020.
51. Jerkovits, T.; Günlü, O.; Sidorenko, V.; Kramer, G. Nested Tailbiting Convolutional Codes for Secrecy, Privacy, and Storage. ACM Workshop Inf. Hiding Multimedia Security; 2020; pp. 79-89.

 doi:10.20944/preprints202011.0209.v1