Efficient key generation scheme for SRAM-PUFs using polar codes

Efficient Key Generation Scheme for SRAM-PUFs using Polar Codes

Bin Chen Tanya Ignatenko Frans M.J. Willems
Eindhoven University of Technology
Dept. Electrical Engineering, SPS Group
P.O. Box 513, 5600 MB, Eindhoven
b.c.chen@tue.nl t.ignatenko@tue.nl f.m.j.willems@tue.nl

Abstract

Physical unclonable functions (PUFs) are a new promising means to realize cryptographic scenarios such as identification, authentication and secret key generation. PUFs avoid the need for key storage, because the device-unique randomness can be translated into a cryptographic key. SRAM-PUFs enjoy the properties that, while being easily evaluated (after a device power-up), they are unique, reproducible, physically unclonable and unpredictable. Error correction codes (ECCs) are essential blocks of secret-generation schemes, since PUF observations are always effected by noise and environmental changes. In this paper, we propose practical error correction schemes for PUF-based secret generation that are based on polar codes. The proposed scheme could generate a 128-bit key or 256-bit key using less PUF bits and helper data bits than before and achieve a low failure probability for a practical SRAM-PUFs application with error probability between 15% and 25%. Therefore SRAM-PUFs are considered to combine very well with authentication and unique cryptographic key generation for resource constrained devices.

1 Introduction

Physical unclonable functions (PUFs) are low-cost hardware intrinsic security primitives that possess an intrinsic randomness (unique "*fingerprint*" for chips) due to the inevitable process variations during manufacturing. Therefore, PUFs can be used to realize cryptographic scenarios that require random, unique and unpredictable keys, such as identification, authentication and cryptographic key generation [1, 2]. PUFs can act as trust anchors and avoid the need for key storage, since the device-unique randomness can be translated into a cryptographic key.

SRAM-PUFs are one of the most popular PUF constructions because they are easy to manufacture and do not require extra investments. SRAM-PUFs also enjoy the properties that, while being easily evaluated (after a device power-up), they are unique, reproducible, physically unclonable and unpredictable [3]. However, SRAM-PUFs cannot be straightforwardly used as cryptographic keys, since their observations are not exactly reproducible due to environmental condition changes such as time, temperature, voltage and random noise. Therefore, error correction techniques are necessary to mitigate these effects and generate reliable keys.

We present a new and efficient key generation building block for SRAM-PUFs key generation, which uses polar codes because their advantage of achieving capacity with low encoding and decoding complexity. To guarantee the performance in terms of reliability and security, and to decrease the required memory size of this scheme, we exploit the efficient decoding algorithm and provide a zero-leakage proof for the proposed scheme. Our simulation results show that 10^{-9} failure probability can be achieved with less PUF cells and helper data bits than before.

2 Secret-Key Generation Model for SRAM-PUFs

2.1 SRAM-PUFs model

SRAM-PUFs are PUF constructions based on the power-up state of an SRAM array. The cell values of an SRAM array after power up go into one of two states: 0 or 1. It has been experimentally demonstrated [4] that due to the independent random nature of process variations on each SRAM cell, the SRAM cell power-up vector can be regarded as a chip fingerprint, see Fig. 1, which is unique and unclonable, and therefore also called physical(ly) unclonable function (PUF). Therefore in this paper we assume that SRAM-PUFs are binary-symmetric, hence for enrollment and authentication PUF pairs (X^N, Y^N) it holds that

$$\Pr\{(X^N, Y^N) = (x^N, y^N)\} = \prod_{n=1}^{N} Q(x_n, y_n), \tag{1}$$

where $Q(0,1) = Q(1,0) = p/2$ and $Q(0,0) = Q(1,1) = (1-p)/2$ and $0 \leq p \leq 1/2$.

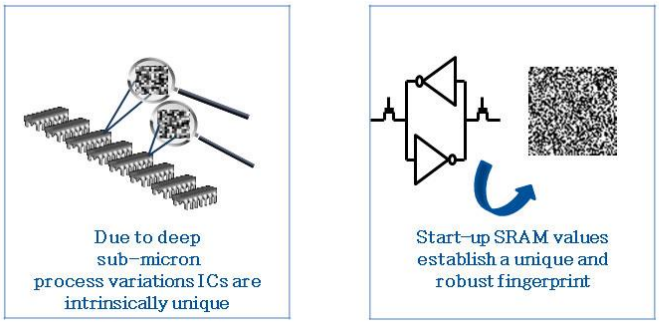

Figure 1: An SRAM based Physical Unclonable Function

2.2 SRAM-PUFs based Secret-Key Generation System

The secret-key generation model can be depicted as a chosen key S sharing scheme between an encoder and a decoder, see Fig. 2. Both parties observe the same SRAM-PUF, resulting in binary observation vectors X^N and Y^N respectively, corresponding to the start-up values of the N cells of SRAM. Therefore, we can consider Y^N as a noisy version of X^N with average bit error probability p. First, the encoder observes an initial enrollment measurement X^N and the secret S, then produces the helper data W. The helper data is assumed to be publicly available for the decoder that also observes the PUF verification sequence Y^N. This decoder forms an estimate $\hat{S}$ of the chosen secret.

In this system we need to design an efficient error correction scheme that ensures the system reliability and security, which indicates that the error probability $\Pr\{\hat{S} \neq S\}$ should be small and moreover that the helper data should not leak any information about the key, i.e. $I(S;W) \approx 0$.

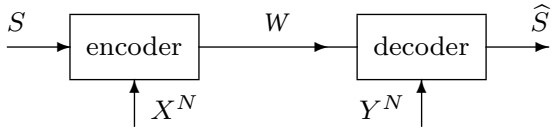

Figure 2: Generic Secret-Key Generation System

3 Polar Codes

Polar codes were proposed by E. Arikan in 2009 [5], who demonstrated how to construct a polar encoder and decoder for any block length N that is a power of 2 and any $K \leq N$. They were the first efficient encoders and decoders proven to achieve the capacity of any binary memoryless symmetric (BMS) channel. Polar codes also provide a flexible selection of the code rate and an arbitrary code rate can be used without re-constructing the code. In this section, we will give an overview of polar codes and then present secret-key generation schemes based on polar codes in next section.

3.1 Channel Polarization

The technique underlying polar codes is "channel polarization"[5] which is an operation that polarizes all sub-channel's mutual informations either to a perfect channel $I(X;Y) = 1$ or to a completely noisy channel $I(X;Y) = 0$.

In Fig. 3, we see $N = 2$ successive channels W_2^1 and W_2^2 that are characterized by transformations $W^- : \mathcal{X} \to \mathcal{Y}^2$ and $W^+ : \mathcal{X} \to \mathcal{Y}^2 \times \mathcal{X}$, respectively. $\oplus$ represents the modulo two sum or equivalently the exclusive "or" operator.

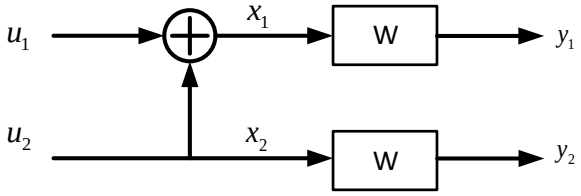

Figure 3: Basic channel transformations

Based on these basic channel transformations, W^- and W^+ can be defined by the following transition probabilities

$$P_{W^-}(y_1, y_2|u_1) = \sum_{u_2 \in \mathcal{X}} \frac{1}{2} P(y_1|u_1 \oplus u_2) P(y_2|u_2), \tag{2}$$

$$P_{W^+}(y_1, y_2, u_1|u_2) = \frac{1}{2} P(y_1|u_1 \oplus u_2) P(y_2|u_2). \tag{3}$$

The following properties related to the above transformation are derived:

1. The mutual information is preserved:

$$I(W^-) + I(W^+) = 2I(W) \tag{4}$$

2. While the channel W^+ is improved, the channel W^- is worsened:

$$I(W^-) \leq I(W) \leq I(W^+) \tag{5}$$

Then, we apply the same basic channel transformation by doubling the numbers N of channels recursively. The capacity of each sub-channel now approaches either 1 (noiseless channel) or 0 (pure noise channel). The transformation matrix G_N is defined by a simple recursive rule,

$$G_N \triangleq R_N G_2^{\otimes n} \tag{6}$$

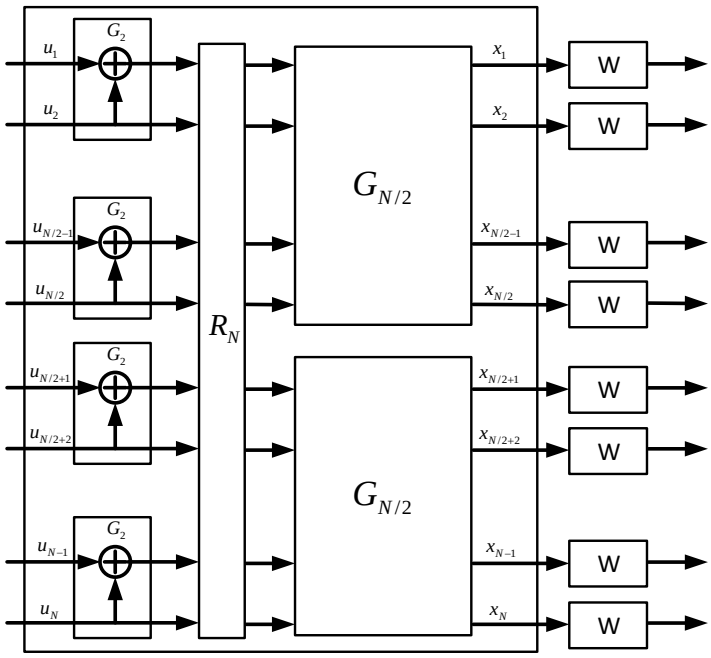

Figure 4: Recursive construction of G_N and G_{2N}

where R_N is a permutation matrix known as *bit-reversal*, $G_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ and the Kronecker power $G^{\otimes n}$ is defined as $G \otimes G^{\otimes(n-1)}$.

In general, G_N is constructed using $N/2$ copies of G_2 and two copies of $G_{N/2}$. The corresponding channel configuration is drawn in Fig. 4 by combining independent copies of W.

3.2 Code Design of Polar Codes

Based on the idea of channel polarization, the recursive operation can create longer channel codes. As the length $N \to \infty$, the error rate of each bit approaches 1 (perfectly reliable) or 0 (completely unreliable). Therefore, polar coding is a strategy to assign the information bits for the reliable channels and set one set of bits with fixed values (1s or 0s) – Arikan calls these the frozen bits – for unreliable channels.

As a family of linear block codes, a binary polar code are can be specified by $(N, K, \mathcal{F}, u^{\mathcal{F}})$, where N is the block length, K is the code dimension (number of information bits encoded per codeword), $\mathcal{F}$ is a set of indices for the $N - K$ frozen bits positions from $\{0, 1, ..., N-1\}$ and $u^{\mathcal{F}}$ is the vector of $N - K$ frozen bits. The frozen bits are normally set to 0, but they may have any value that is known to both the encoder and the decoder. The optimized polar codes for target channel transition matrix W can be constructed by choosing $\mathcal{F}$ as the set of inputs with the lowest error probabilities. Therefore, the choice of the set $\mathcal{F}$ is a critical step for polar coding often referred to as *polar code construction*. The original construction of polar codes is based on the Bhattacharyya bound approximation [5]. Later proposed algorithms improve on this approximation at the cost of higher complexity [6, 7, 8].

3.3 Encoding

For an $(N, K, \mathcal{F})$ polar code, the encoding operation for vector of information bits $\mathbf{u}$ can be represented using a generator matrix,

$$\mathbf{G_N} = \mathbf{G}^{\otimes \mathbf{N}}, \tag{7}$$

where $\otimes$ denotes the Kronecker product. Given the data sequence $\mathbf{u}$, the codewords are generated as

$$\mathbf{x} = \mathbf{G}_{\mathcal{F}^c}\mathbf{u} + \mathbf{G}_{\mathcal{F}}\mathbf{u}^{\mathcal{F}}, \tag{8}$$

where $\mathcal{F}^c \triangleq \{0, 1, ..., N-1\}\backslash\mathcal{F}$ corresponds to the non frozen bits indices. $\mathbf{u}$ is the data sequence, and $\mathbf{u}^{\mathcal{F}}$ is the sequence of frozen bits which we set as all zeros.

3.4 Decoding

At the decoder, we want to decode the output of the N channels which defined by the transition probabilities $P_i(y_1^N u_1^{i-1}|u_i)$. Therefore, we need the correct estimates of the previous channel inputs $\hat{u}_1, ..., \hat{u}_{i-1}$ to estimate the channel input $\hat{u}_i$. Based on this, it is more suitable to use a successive cancellation decoder. Given y_1^N and estimates $\hat{u}_1^{i-1}$ of u_1^{i-1}, the SC decoding algorithm attempts to estimate u_i. This can be implemented by computing the log-likelihood ratios $L_N^i(y_1^N, \hat{u}_1^{i-1}) = \log \frac{Pr(0|y_1^N, u_1^{i-1})}{Pr(1|y_1^N, u_1^{i-1})}$, where the LLRs can be computed recursively using two formulas:

$$\begin{aligned} L_N^{2i-1}(y_1^N, \hat{u}_1^{2i-2}) = 2\tanh^{-1}(&\tanh(L_{N/2}^i(y_1^{N/2}, \hat{u}_o^{2i-2} \oplus \hat{u}_e^{2i-2})/2) \\ &\cdot \tanh(L_{N/2}^i(y_1^{N/2+1}, \hat{u}_e^{2i-2})/2)), \end{aligned} \tag{9}$$

and

$$L_N^{2i}(y_1^N, \hat{u}_1^{2i-1}) = (-1)^{\hat{u}_{2i-1}} L_{N/2}^i(y_1^{N/2}, \hat{u}_o^{2i-2} \oplus \hat{u}_e^{2i-2}) + L_{N/2}^i(y_1^{N/2+1}, \hat{u}_e^{2i-2}), \tag{10}$$

where $\hat{u}_o^{2i-2}$ and $\hat{u}_e^{2i-2}$ denote, respectively, the odd and even indices part of $\hat{u}^{2i-2}$. Therefore, calculation of LLRs at length N can be reduced to calculation of two LLRs at length $N/2$, and then recursively break down to block length 1. The initial LLRs can be directly calculated from the channel observation as $L_1^i = \log \frac{Pr\{0|y_i\}}{Pr\{1|y_i\}}$.

For finite block-length, the performance of polar codes can be improved by implementing enhanced decoding algorithms based on the classical successive cancellation decoder (SCD), such as successive cancellation list (SCL) decoding [9] and CRC-aided successive cancellation list (CA-SCL) decoding [10].

4 Secret-Key Generation Schemes based on Polar Codes

4.1 Polar Codes based Code-offset Construction

In this section, we show how helper data can be constructed using polar codes for a PUF-based key generation scheme. Fig. 5 illustrates the polar code based code-offset construction scheme that realizes an enrollment phase (encoder) and key regeneration phase (decoder).

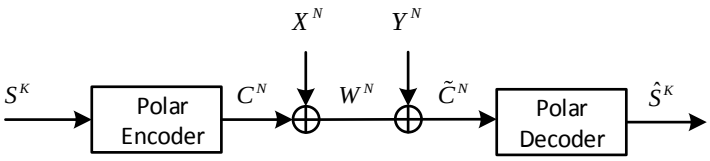

Figure 5: Polar codes based code-offset scheme.

4.1.1 Enrollment

In the enrollment phase, the secret key S^K is encoded into a polar codeword $C^N = p(S^K) = \mathbf{G}_{\mathcal{F}^c}\mathbf{S^K} + \mathbf{G}_{\mathcal{F}}\mathbf{0^{N-K}}$, where $p(\cdot)$ is the polar encoding function, $\mathcal{F}^c$ and $\mathcal{F}$ are the index sets for secret key and the frozen bits. Then, the helper data is generated by adding the PUF enrollment response sequence X^N to this codeword.

$$W^N = C^N \oplus X^N = p(S^K) \oplus X^N. \tag{11}$$

4.1.2 Key regeneration

In the key regeneration phase, a PUF authentication sequence Y^N is observed and added to the public helper data W^N. We obtain the cordword with noise $e^N = X^N \oplus Y^N$,

$$\widetilde{C}^N = W^N \oplus Y^N = p(S^K) \oplus \underbrace{(X^N \oplus Y^N)}_{e^N}. \tag{12}$$

Hence, the secret key S^K can be estimated by implementing a modified version of the SC and CA-SCL decoding algorithms with a known $N-K$ zeros-vector as

$$\hat{S}^K = \mathrm{SCD}(C^N, \mathbf{0}^{N-K}), \quad \hat{S}^K = \mathrm{SCLD}(C^N, \mathbf{0}^{N-K}). \tag{13}$$

where the polar decoder $\mathrm{SCD}(\cdot)$ is given in Algorithm1 and $\mathrm{SCLD}(\cdot)$ is the polar decoder with CRC-aided SCL decoding algorithms of [10].

Algorithm 1 Decoding Algorithm for Code-offset Construction

Input: The observations Y^N from PUFs, the public helper data W^N.
Output: The estimated secret $\hat{S}^K$

Compute the initial $L_1(y_i) = \log \frac{Pr\{0|y_i\}}{Pr\{1|y_i\}}, i = 1, 2, ...N$
Compute $L_1(\widetilde{c}_i) = (-1)^{w_i} L_1(y_i), i = 1, 2, ...N$
for $i = 1$ to N **do**
  Compute $L_N^i(\widetilde{C}^N, \hat{u}_1^{i-1})$ with the initial $L_1(\widetilde{C}^N)$ from Eq. (9-10)
  if $i \in \mathcal{F}$ **then**
    $\hat{U}_i = 0$
  else if $i \in \mathcal{F}^c$ and $L_N^i(\widetilde{C}^N, \hat{u}_1^{i-1}) > 0$ **then**
    $\hat{U}_i = 0$
  else
    $\hat{U}_i = 1$
  end if
  $\hat{S}^K \leftarrow \hat{U}^N[\mathcal{F}^c]$
end for
return $\hat{S}^K$

4.2 Security Analysis

In this section, we analyze the secrecy for the proposed polar codes based key generation scheme. To prove that the helper data do not leak any information about the secret key, it requires that the helper data leaks about the generated secret key. Therefore we must show that $I = (S^K; W^N) = 0$. For the proposed scheme, we can easily prove that this is true each pair of secret S^K and helper data W^N, see below:

$$\begin{aligned} 0 \leq I(S^K; W^N) &= I(C^N; W^N) = H(W^N) - H(W^N|C^N) \\ &= H(C^N \oplus X^N) - H(C^N \oplus X^N|C^N) \\ &\leq N - H(X^N|C^N) = N - H(X^N) = N - N = 0. \end{aligned} \tag{14}$$

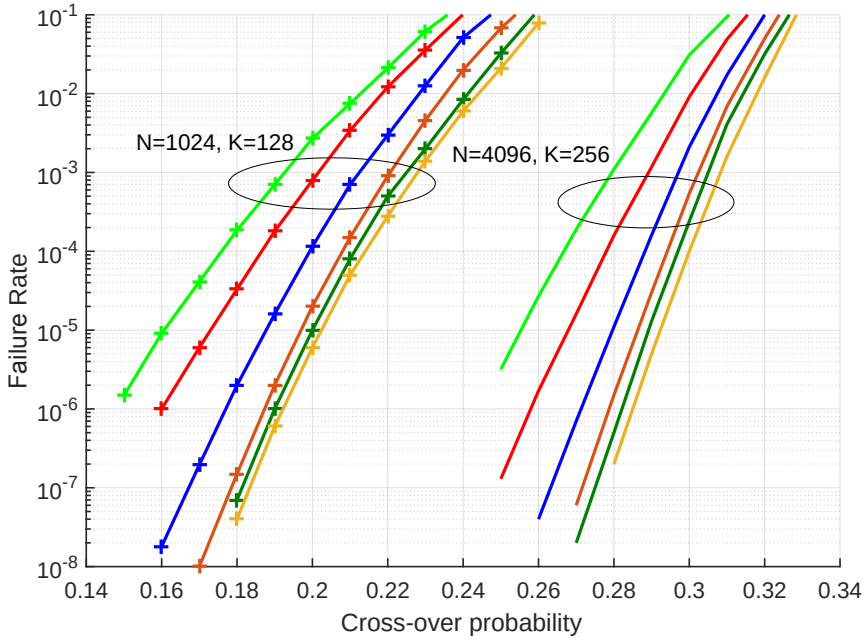

Figure 6: Failure rate performance comparison of the polar codes based secret-key generation scheme with different key sizes $K =\in \{128, 256\}$ and different list sizes $L \in \{1, 2, 4, 8, 16, 32\}$ for CA-SCL decoding.

5 Simulation Results

In this section, we present the performance of polar code based error correction schemes for an SRAM-PUF with two average bit error probability around 15% and 25%, respectively. For these two different conditions, we designed the corresponding polar code constructions to achieve the target performance.

The most important performance criterion for PUF error correction is the error probability (or failure probability) of the key regeneration. Fig. 6 gives the error correction performance of the polar code based code-offset construction with the SC and CA-SCL decoding algorithm. Here SC and CA-SCL with different required key size are simulated for different practical applications.

For key size $K = 128$ with block length $N = 1024$, we can see that the failure rate of polar codes with SC decoding is close to 10^{-6} at 15% and CA-SCL decoding can further reduce the failure rate to less than 10^{-9} at 15% as list size L increases, but at the cost of extra computational complexity and memory. For key size $K = 256$ with block length $N = 4096$, the simulation results show that failure rates 10^{-6} and 10^{-9} can also be achieved under worse PUFs condition $p = 25\%$.

6 Conclusion

In this paper we investigated practical polar code based code-offset schemes for secret-key generation scheme that encode the secret key into polar codeword and then mask it with PUF bits as helper data to correct the errors for the noisy PUFs. Our simulation results show that reliability (small failure probabilities) can be achieved together with high security. Furthermore, the proposed scheme requires less SRAM-PUF bits and helper data bits, which leads to the reduction in memory requirements.

Acknowledgment

This work was funded by Eurostars-2 joint programme with co-funding from the EU Horizon 2020 programme under the E! 9629 PATRIOT project.

References

[1] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *2007 44th ACM/IEEE Design Automation Conference*, June 2007, pp. 9–14.

[2] S. U. Hussain, M. Majzoobi, and F. Koushanfar, "A built-in-self-test scheme for online evaluation of physical unclonable functions and true random number generators," *IEEE Trans. on Multi-Scale Computing Systems*, vol. 2, no. 1, pp. 2–16, Jan 2016.

[3] R. Maes and I. Verbauwhede, *Physically unclonable functions: a study on the state of the art and future research directions*, 2010, pp. 3–37.

[4] G.-J. Schrijen and V. van der Leest, "Comparative analysis of SRAM memories used as PUF primitives," in *Conf. on Design, Automation and Test in Europe*, San Jose, CA, USA, 2012, pp. 1319–1324.

[5] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051–3073, July 2009.

[6] R. Mori and T. Tanaka, "Performance and construction of polar codes on symmetric binary-input memoryless channels," in *IEEE Int. Symp. Inf. Theory*, June 2009, pp. 1496–1500.

[7] S. Zhao, P. Shi, and B. Wang, "Designs of Bhattacharyya parameter in the construction of polar codes," in *7th Int. Conf. on Wireless Comm. Networking and Mobile Computing*, Sept 2011, pp. 1–4.

[8] I. Tal and A. Vardy, "How to construct polar codes," *IEEE Trans. Inf. Theory*, vol. 59, no. 10, pp. 6562–6582, Oct 2013.

[9] ——, "List decoding of polar codes," in *IEEE Int. Symp. Inf. Theory*, July 2011, pp. 1–5.

[10] ——, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, May 2015.