REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
07-08-2023	Final Report	1-Nov-2015 - 31-Jul-2016

4. TITLE AND SUBTITLE

Final Report: Research Area 11: ARO Special Programs: Building Robust and Practical PUFs with Configurable Ring Oscillators

5a. CONTRACT NUMBER
W911NF-15-1-0289

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
611102

6. AUTHORS

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES

University of Maryland - College Park
The University of Maryland
Office of Research Administration
College Park, MD 20742 -5141

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)

U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
66419-NC-II.1

12. DISTRIBUTION AVAILIBILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Gang Qu
UU	UU	UU	UU		19b. TELEPHONE NUMBER: 301-405-6703

RPPR Final Report
as of 07-Aug-2023

Agency Code: 21XD

Proposal Number: 66419NCII **Agreement Number: W911NF-15-1-0289**
INVESTIGATOR(S):

Name: Gang Qu
Email: gangqu@mail.umd.edu
Phone Number: 3014056703
Principal: Y

Organization: **University of Maryland - College Park**
Address: The University of Maryland, College Park, MD 207425141
Country: USA

DUNS Number: 790934285 EIN: 526002033
Report Date: 31-Oct-2016 Date Received: 07-Aug-2023
Final Report for Period Beginning 01-Nov-2015 and Ending 31-Jul-2016
Title: Research Area 11: ARO Special Programs: Building Robust and Practical PUFs with Configurable Ring Oscillators
Begin Performance Period: 01-Nov-2015 **End Performance Period:** 31-Jul-2016
Report Term: 0-Other

Submitted By: Gang Qu Email: gangqu@mail.umd.edu
Phone: (301) 405-6703

Distribution Statement: 1-Approved for public release; distribution is unlimited.

STEM Degrees: **STEM Participants:**

Major Goals: see the uploaded report.

Accomplishments: see the uploaded report.

Training Opportunities: One MS student and two Ph.D. students were partially supported by this grant.

Results Dissemination: see the uploaded report.

Honors and Awards: Nothing to Report

Protocol Activity Status:

Technology Transfer: Nothing to Report

PARTICIPANTS:

Participant Type: PD/PI
Participant: Gang Qu
Person Months Worked: 1.00 **Funding Support:**
Project Contribution:
National Academy Member: N

RPPR Final Report
as of 07-Aug-2023

Partners

,

I certify that the information in the report is complete and accurate:
Signature: Gang Qu
Signature Date: 8/7/23 3:47PM

Project Title: Building Robust and Practical PUFs with Configurable Ring Oscillators
Performance Period: 11/1/2015 – 7/31/2016

Abstract

Physical Unclonable Function (PUF) is one of the most promising hardware features that can be utilized to improve system security. Ring Oscillator (RO) PUF captures the delay difference of a pair or a group of ROs introduced during fabrication process and defines PUF secrecy based on such difference. In this project, we propose a framework to build RO PUF at inverter level, instead of RO level. This will provide us the flexibility in choosing whether an inverter should be included in the RO-PUF and hence improve the security and reliability of RO PUF secrecy and reduce its hardware cost. We anticipate the successful completion of this project will have direct impact on the hardware related design for trustworthy computing, which is of critical importance for cyber warfighting. Silicon PUF, despite its great promise in enhancing system's security and trustworthiness, will not be adopted in real systems unless its high hardware inefficiency and unreliability issues are solved. This research will provide a timely solution to these problems and thus will have huge impact on security and trustworthy computing.

Objectives

We propose to investigate the delay measurement scheme at inverter level with special focus on (1) how to minimize the number of physical measurements we have to take to guarantee the accuracy of the calculated delay of each delay unit; and (2) what is the delay accuracy we have to achieve to ensure that the configurable RO PUF will generate reliable PUF bits when operating environment changes. *Another goal of this proposal is to study how to build PUFs based on the relatively accurate delay information of each inverter in order to improve PUF's usability in terms of hardware efficiency, information security, and information reliability.*

Findings

Our proposed flexible RO PUF has the fundamental difference with the existing PUF approaches. In our design, after the chip is fabricated, we can measure the real delay difference of each stage. Then construct the ring oscillator through pre-computation. Moreover, compared with traditional RO PUF, which only has the bit-stream output, our design contains the configuration vectors as additional information. Keeping this difference in mind, we can facilitate new security properties. In [1], we demonstrate that the configuration vectors can help to prevent cloning. The vectors have the mapping relation with the secret keys. Compared with the traditional PUF, this relation makes the cloning more difficult. The potential cloning method doesn't work for our design. The vectors are generated through computation based on the measurement. So the rule of mapping is determined rather than randomly generated. Thus the configuration vectors should be protected from being tampered with.

Our single RO PUF uses only one RO to generate one PUF bit [1]. This offers several advantages. Because the fast sub-ring and slow sub-ring share the same path, and the value of the PUF bit will be determined by whether a specific inverter belongs to the fast sub-ring or the slow sub-ring, the on-chip spatial variation will have less impact and we can expect more reliable PUF bit. Also the path sharing makes this PUF resilient to EM measurement based side-channel attacks.

As the follow-up of this funded research, we showed in [2] that the proposed RO PUF can be built against advanced machine learning attacks; we also showed in [3] that our RO PUF can be used to enhance the entropy of random entropy sources, particularly when the original entropy source does not provide high entropy [3] and applied this to improve the password security and usability for IoT applications.

PDF of the four references are attached in this report.

[1] M Gao, K Lai, J Zhang, G Qu, A Cui, and Q Zhou. Reliable and anti-cloning PUFs based on configurable ring oscillators. 14th International Conference on Computer-Aided Design and Computer Graphics. 2015.

[2] Q Wang, M Gao, and G Qu. A machine learning attack resistant dual-mode PUF. Proceedings of the Great Lakes Symposium on VLSI, 177-182, 2018.

[3] Q Wang and G Qu. A silicon PUF based entropy pump. IEEE Transactions on Dependable and Secure Computing 16 (3), 402-414, 2018.

[4] Q Wang, M Gao, and G Qu. Puf-passse: A puf based password strength enhancer for iot applications. 20th International Symposium on Quality Electronic Design (ISQED), 198-203, 2019.

2015 14th International Conference on Computer-Aided Design and Computer Graphics

Reliable and Anti-Cloning PUFs based on Configurable Ring Oscillators

Mingze Gao* Khai Lai†, Jiliang Zhang‡, Gang Qu*, Aijiao Cui§, Qiang Zhou¶
*Department of Electrical and Computer Engineering and Institute for Systems Research,
University of Maryland, College Park, USA
†Availink Inc, MD, USA
‡Software College, Northeastern University, Shenyang 110819 China
§School of Electronic and Information Engineering,
Harbin Institute of Technology Shenzhen Graduate School, China
¶Department of Computer Science and Technology, Tsinghua University, Beijing, P.R. China
{{mgao1, gangqu}@umd.edu, lai.khai@yahoo.com, zhangjl@swc.neu.edu.cn,
cuiaj@hitsz.edu.cn, zhouqiang@tsinghua.edu.cn,}

***Abstract*—Ring oscillator Physical Unclonable Function (RO PUF) is a popular silicon PUF due to its ease of implementation on both ASIC and FPGA. However, RO PUFs have severe reliability issues when the operating environment deviates from the nominal condition and security issues as cloning attacks have been reported. In this work, we propose to build configurable RO PUFs based on the notions of configurable RO PUF [6,16] and highly flexible RO PUF [22] to address these concerns. First, we demonstrate how to build RO PUF from single flexible ROs, which improves both the reliability and hardware efficiency of RO PUFs. Then we propose a novel dual voltage based configurable RO PUF to mitigate the cloning attacks. Our experimental results show that our configurable RO PUFs are more reliable and hardware efficient than the existing RO PUF designs. Using the flexible RO PUF [22] as baseline, we have reduced the bit flip rate by 69% and improve the hardware utilization by 136%. In addition, the anti-cloning approach generates PUF data significantly different from the original PUF secret (average 47.5% Hamming distance) which makes potential cloning attacks very difficult.**

***Keywords*-ring oscillator; PUF; configurable; reliability; voltage scaling; anti-cloning**

I. Introduction

Ring oscillator (RO) PUF is a popular silicon PUF that can generate highly reliable outputs by amplifying the delay difference caused by fabrication variations through the RO substructure. For two ring oscillators RO_1 and RO_2, we can measure their amplified delay difference by connecting them to two counters and then compare the readings of the counters. We can define that, for instance, a bit '1' is generated if RO_1 is faster than RO_2 and that a bit '0' is generated otherwise. However, with the working environment changes, especially temperature and supply voltage, the delay difference will not always remain the same. This problem severely challenges the PUF's reliability requirement. If the impact is sufficiently large, it becomes possible that one RO is faster than the other at one voltage or temperature but becomes slower at another voltage or temperature. If this is the case, the bit generated from this pair of ROs may flip and cannot be used for security applications.

One of the most important and effective methods to improve PUF reliability is to increase the delay difference between the pair of ROs [3,6-8]. For example, the configurable RO PUF provides two inverters for selection at each stage [6] and the highly flexible RO PUF only includes selective inverters instead of using all the inverters [22]. In this paper, we propose a configurable RO PUF based on the concept of flexible RO PUF to answer some of the tough challenges in RO PUF design. More specifically, we first combine the concepts of configurability and flexibility to build a new RO PUF that is capable of generating one PUF bit from each single RO. This is a novel structure of RO PUF because it does not require delay comparison between a pair of a group of ROs. Therefore, it can lead to promising hardware efficiency. Second, we leverage the sensitivity of delay to supply voltage change and configure the RO PUF carefully such that the PUF bits remain reliable at normal voltage levels, but will become different when we change voltage to a level higher than normal. This feature can be used to countermeasure the recently potential cloning attacks [1,14]. Experimental results demonstrate that the configurable RO PUFs can deliver these promises. We believe that more security applications can be found when we fully utilize the concepts of configurability and flexibility. The rest of the paper is organized as follows. We briefly survey the current research on RO PUF in Section II. The configurable and flexible RO PUFs are introduced in Section III. In Section IV and V, we demonstrate how to combine them to improve reliability, hardware efficiency, and security (anti-cloning attacks). Section VI reports experimental results.

II. Related Work

Since the first introduction of the notion of PUF, a variety of PUF hardware structures has been proposed [23], among

This work is supported in part by the National Natural Science Foundation of China under Grant No. 61228204 and 61176035.

978-1-4673-8020-1/15 $31.00 © 2015 IEEE
DOI 10.1109/CADGRAPHICS.2015.54

which silicon PUFs are of the most interest in terms of fabrication cost and readiness to be integrated into tamper-resistant devices. This rapid development successfully brings the PUF into several security-related application, such as the IP protection [24]. Generally speaking, silicon PUFs exploit the uncontrollable wire delay and device voltage transfer characters determined by fabrication process variations. Such unique characteristics of each fabricated IC are measurable through embedded ring oscillators [3], multiplexer (arbiter)-based delay circuits [21] or cross-coupled circuits [4], or the random fluctuations of SRAM cells [2,5]. Different PUF structures determine that one PUF may have advantages over others under certain design constraints. For instance, arbiter PUF is more suitable for resource-constrained applications than RO PUF [3]. However, RO PUF provides higher reliability over arbiter PUF or Butterfly PUF [4] under a wide range of temperature fluctuation.

Due to their different mechanisms, RO PUF has arguably the highest requirement on hardware efficiency than other silicon PUFs. With n ROs, the original pair-wised RO PUF [3] can generate $n/2$ PUF bits, however, as we have mentioned earlier, these bits are not reliable. The 1-out-of-8 scheme [3] significantly improves the reliability at the cost of hardware efficiency. It can only produce $n/8$ PUF bits. Maiti and Schaumont [8] suggest pairing up the adjacent ROs, which increases the PUF bits from $n/8$ to about $n-1$. Another concern of RO PUF is the data dependency in the PUF information extracted from the ROs.

Meanwhile, there are recent reports on the vulnerabilities of PUF and several successful attempts to clone the unclonable PUF information. Karakoyunlu and Sunar [9] reported the first successful power side-channel attack on the software implementation of fuzzy extractor. This implies that software implementation of any error correction scheme is potentially vulnerable to side-channel attack and error correction can lower the security of PUF. Merli et al. [10] studied the side-channel analysis of silicon PUFs and their fuzzy extractors. They pointed out that the frequencies of ROs on FPGA can be measured with state-of-the-art EM equipment and thus it becomes possible to clone the RO PUF. They also implemented attacks on the fuzzy extractor which can successfully extract the cryptographic keys generated by PUFs using fuzzy extractor. In another work [11], the same authors showed that by exploiting the chained challenges and EM emanation, it is possible to deduce the relative frequency rank of the ROs and guess correctly the PUF secret bits. More recently, the same group [12] demonstrated that it is feasible to measure the EM emission of a single tiny RO with only three inverters within a single configurable logic block on an FPGA chip. Helfmeier et al. [13] demonstrated the first successful cloning of an SRAM PUF based on the fact that SRAM cells emit near infrared light when it is read and the cells power-up value can be obtained from the emitted light. Rhrmair et al. [14] described how to use machine learning techniques to attack both RO PUF and arbiter PUF by modeling the PUF behavior. Mahmoud et al. [15] combined the machine-learning based modeling techniques and side channel information leak to attack strong PUFs such as the XOR arbiter PUF.

III. Configurable and Flexible RO PUFS

The notion of configurability in RO PUF has been introduced by Maiti and Schaumont[6]. Their main approach is at each stage of the RO, there is a multiplexer selecting one of out of two inverters. For the 3-stage RO shown in Fig. 1, there will be 8 possible configurations. The configurations of one pair of ROs, whose delay difference is the largest, are used for regenerating the PUF output bit. Their 3-stage configurable RO PUF occupies two Configurable Logic Blocks (CLB) in Xilinx FPGAs; each 3-stage configurable RO occupies a single CLB.

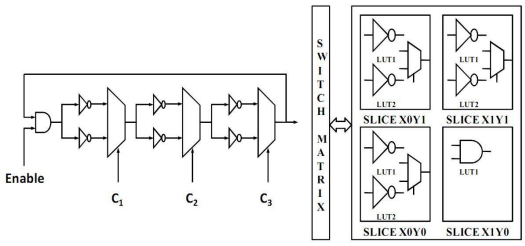

Figure 1. Configurable RO PUF proposed in [6]

Xin et al. [16] provide an improvement by increasing the number of possible configurations to 256 and still using the same number of CLBs. Compared to their works, Gao et al's approach [22] is more flexible because during the construction of each RO, there will be the option to use or not use each of the inverters. In other words, currently "configurable" means that we can choose which inverter to use at each stage, but the number of stages (or the number of inverters) in the RO is fixed; while "flexible" means that the number of stages in an RO can be changed based on the delay variations. In the rest of this section, we introduce the flexible RO proposed in [22].

Fig. 2 depicts the architecture that gives the flexibility to select inverters for the construction of ROs. A multiplexer is added after each inverter to control whether the inverter will be included in the RO. This is achieved by the selection bit of the multiplexer. If the selection bit is '1', the corresponding inverter will be included in the RO; if the selection bit is '0', the inverter will not be used and the signal will go through the wire to the next inverter. We use the term configuration vector to refer the collection of all the multiplexer selection bits.

The delay of inverter can be measured by selecting each inverter individually. The approach in [22] is illustrated below.

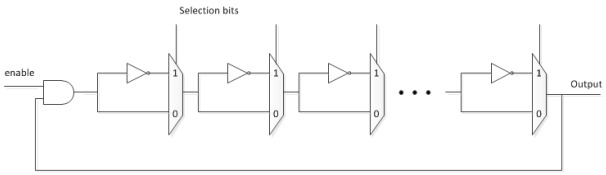

Figure 2. Flexible RO PUF proposed in [22]

Example 1: Consider a pair of ring oscillators, RO_1 and RO_2, each consisting of 7 inverters. Assuming that the delays of these inverters are: {d1=5, d2=4, d3=8, d4=7, d5=4, d6=6, d7=3} and {c1=7, c2=5, c3=7, c4=5, c5=1, c6=4, c7=5}, where di and ci denote the delay of the ith inverter in RO_1 and RO_2, respectively. The total delay of these two ROs can be computed as

$D_{RO1} = 5 + 4 + 8 + 7 + 4 + 6 + 3 = 37$

$D_{RO2} = 7 + 5 + 7 + 5 + 1 + 4 + 5 = 34$

RO_1 is 37-34 = 3 units of time slower than RO_2. This delay difference can be used to generate one PUF bit and its reliability is determined by this delay difference. In general, a large delay difference leads to a reliable bit. In [22], the three inverters in the middle {d4, d5, d6} and {c4, c5, c6} are selected to build RO_1 and RO_2, respectively. The delay difference becomes (7+4+6)-(5+1+4) = 7 units of time, which is more than twice as large as the delay difference when all the inverters are included in the ROs.

The rationale behind this is that the fabrication variation is unpredictable. An inverter in RO_1 is equally likely to be faster or slower than the inverter at the same position in RO_2. When all the inverters are included in the ROs, the delay difference between inverters at the same position can be canceled out. In this example, although RO_1 is slower than RO_2, it has faster inverters at the first, second and last position than RO_2, including these three inverters in the ROs will reduce their delay difference. Selectively choosing inverters to configure the RO can increase the gap between two ROs' total delays and thus improve the PUF bit's reliability.

IV. Reliable and Hardware Efficient RO PUF

Compared with traditional pairwised RO PUF, even though [22] increases the reliability, the hardware efficiency does not get any improvement. According to this disadvantage, in this section, we propose a novel approach for configurable RO to significantly improve the hardware efficiency while maximizing the PUF's reliability. Our work is based on the configurable structure proposed by Gao et al. But our contribution is totally different with theirs. We explain our One-RO-One-Bit (OROB) approach through a motivational example. Then we compare our new approach with the existing configurable ideas.

Motivational Example 2: Let us consider the 3 fastest inverters with the 3 slowest inverters in each RO. For RO_1, the total delay of the 3 fastest ones $D_{fast_1} = 4+4+3 = 11$ and the total delay for the 3 slowest ones is $D_{slow_1} = 8 + 7 + 6 = 21$. We see a delay difference of 10 units of time. For RO_2, we have $D_{fast_2} = 4 + 1 + 5 = 10$, $D_{slow_2} = 7 + 7 + 5 = 19$, and a delay gap of 9 units of time. All the new generated gaps are much bigger than the gap when using all inverters. If we can create a PUF bit by comparing the faster inverters with the slowest inverters in the same RO, we are able to generate bits from both RO_1 and RO_2 that are more reliable than the bit generated by comparing RO_1 and RO_2. More importantly, we get two PUF bits instead of one.

In the traditional RO PUF, the bit is generated not only by the comparison of frequencies or delays, but also two ROs physical locations (the top one and the bottom one). However, in our OROB approach, each bit comes out from the same RO. There is no more top and bottom. Therefore, we define a new bit generation mechanism using the configuration vector. Consider the configurable RO architecture shown in Fig. 1, if the configurable RO has 10 inverters, we can choose 5 fastest ones and 5 slowest ones (we have to choose odd numbers of inverters to make RO oscillate). Their configuration vectors V_1 and V_2 should be complementary to each other, assuming V_1= (1011011000) and V_2= (0100100111). We can define the sub-ring containing the first inverter (the first bit of configuration vector is '1') as the top RO, and then the other is the bottom one. So in the example, V1 represents the top RO, and V_2 represents the bottom RO. '1' can be defined if the V_1 is faster than V_2, otherwise '0'. Compared with other existing approaches, such as traditional pairwised RO PUF, 1-out-of-8 RO PUF, neighbor chainwised RO PUF, our approach dominates in both hardware efficiency and response reliability. Compared with [22], our approach not only maintains the high reliability, but also almost doubles the hardware efficiency.

As the bit is generated from the same RO, the frequencies measurement process through counter cannot be parallelized. Firstly, we enable the top subring $subRO_1$ using configuration V_1. Secondly, we send the frequency value f_1 from counter to secure memory. Then we enable the bottom subring $subRO_2$ using V_2. The comparison is between the new countered f_2 in the counter and the previous stored $f1$ in the secure memory. People may argue the potential information leak of storing and retrieval process of f_1. However, if the attacker knows f_1 through some side channel attacks, there is no help to guess out the PUFs response since f_2 is totally unknown. The configuration vectors have the spatial mapping relation with the response bits, just like the physical location of each RO in traditional PUF. If the attackers hack the configuration vectors, they only know which subRO is top one which is bottom one. Exposure of configuration vector does not detriment the security of

PUF. To get the secrete key, they still need to run the PUF.

The configuration vectors can also have multiple usages. Take chip/device authentication for example, we can release the configuration vectors as a public ID of the chip/device and keep the PUF bits as a secret or private ID. When the chip/device needs to authenticate itself or to be authenticated with low level of security or confidence, such public ID can be used. The secret PUF bits will be reserved for high security authentication applications. In this case, the configuration vectors are also need to protect, such as being stored in secure memory.

V. Anti-Cloning RO PUF

A. Potential clone threat analysis

PUF is short for "Physical Unclonable Function", whose the most important feature is "Unclonable" obviously. But this feature is threatened with the attack technique development. Silicon PUFs are based on fabrication variations. This intrinsic feature cannot be cloned atom by atom due to the limitation of current fabrication technology. However, instead of cloning the exactly same variations, attackers nowadays are putting efforts on constructing the functional clones. The functional clone refers to a clone method guaranteeing that the cloned PUF generates the same response even though its structure might be very different with the original one. This leads us to the risky place where the "unclonable" feature is lost or weakened. The deep reason to make functional clone possible is that we digitize the PUF's variations and omit its analog features. Moreover, when the digitization process is not reliable, namely when bit flip happens, various methods are proposed to physically enlarge the intrinsic fabrication variation to achieve a stable digital output. For example, optical proximity correction method is used to make PUF information unique [18], and several approaches have been proposed to improve the reliability of delay-based PUF under temperature variations [7,19]. However, as a side effect, these techniques make it possible to physically extract the PUF's digitization results, such as using EM emanation to capture the frequency differences if these differences are large enough. When the digitization information is available, cloning the PUF becomes much simpler. In the case of RO PUF, it is trivial to build a RO faster than the other (for example, by adjusting gate size or threshold voltage or simply replacing it with a wire). Here is a simple scenario to prove the feasibility of cloning RO PUF.

Cloning Scenario: There are two pairs of RO $\{$A, B$\}$ and $\{$C, D$\}$ with 5 inverters in each, generating two bits 01. This means $delay.A > delay.B$, and $delay.C < delay.D$. The attackers can use the EM emanation to measure these two relations. To clone this PUF, they can simply build A and D with 5 inverters and build B and C with 1 inverter. The inverter numbers' mismatching will guarantee the clonable PUF generate the same response with the original one.

B. Authentication based Anti-clone Approach

The traditional RO PUF does not have the resistance to the above scenario. However, using configurable RO, we can prevent or detect this potential clone threat with the help of configuration vectors. The idea is alternated by authentication process. Besides the secret key configuration vector V_k, we introduce the testing vector V_t for anti-cloning purpose. Testing vector V_t is carefully selected to generate stable testing response R_t. In the working phase, V_k is configured to PUF to obtain the reliable secret key. In detecting phase, V_t is configured to PUF to achieve the test bitstream R_t. For cloning detection, we can use multiple testing vectors. All these vectors should be very different with each other. It is better to cover all the inverters during the testing phase. In the cloning scenario, it could possibly provide the same response under a specific configuration vector. However, when using several configuration vectors, the probability of cloning all the right responses decrease dramatically.

The rationale of above approach is that configurable RO can be used to generate Challenge-Response-Pairs (CRPs). The configuration vector is the challenge. The famous 1-out-of-8 RO PUF and [6] provide the Challenge and Response schematics, respectively. But the sizes of their challenges are not large enough for authentication. Configurable RO PUF can provide enough CRPs with adequate length of each RO. Even though configurable RO PUF can be used in authentication field, we do not suggest it to be a conventional authentication PUF, like arbiter PUF. People should not get unlimited access of V_t, otherwise it will suffer from modeling attack [1]. A few carefully selected configuration vectors are enough to reduce the cloning risk.

C. Dual Voltage based Anti-cloning Approach

The lesson we have learned in above section is that the intrinsic fabrication variation of silicon PUFs is unclonable, but PUF is clonable during the process of digitization that losing the distinctive analog information. Since digitization is inevitable in the digital circuit, we propose a novel anti-cloning mechanism to extract more utilization of PUF's physical characters during the digitization process. The following motivational example will illustrate how our configurable RO PUF can use a voltage scaling scheme to detect and thus defeat physical cloning.

Motivational Example 3: Consider the following two ROs, each consisting of 5 inverters whose delays are:

At voltage V_1:

$RO_1 = \{4, 5, 3, 7, 6\}$ $RO_2 = \{6, 3, 4, 7, 4\}$

At voltage V_2 ($V_2 < V_1$):

$RO_1 = \{10, 12, 9, 13, 11\}$ $RO_2 = \{11, 8, 10, 12, 9\}$

As the supply voltage decreases, the delays of the inverters increase. Because of the configurable feature, we don't have to use all the inverters. In this example, we can choose the first three. At V_1, $delay.RO_1 = 4 + 5 + 3 = 12$,

$delay.RO_2 = 6{+}3{+}4 = 13$, and $delay.RO_1 < delay.RO_2$. But when we change the voltage to V_2, $delay.RO_1 = 10 + 12 + 9 = 31$, $delay.RO_2 = 11 + 8 + 10 = 29$, and so $delay.RO_1 > delay.RO_2$. So if at V_1 the PUF output bit is "0", the bit will flip to "1" when the supply voltage changes to V_2. Prior to the PUF secret key generation phase, the configuration vector of RO_1 and RO_2 is set to $\{11100\}$. We can get bitstreams S_1 and S_2 under the working voltage V_1 and V_2 respectively. If S_1 is complementary to S_2, the PUF's authenticity is confirmed. Moreover, because the usages of controlled reversible bitstream $\{S_1, S_2\}$ are to verify and prevent cloning, they do not need to be exactly complementary to each other. Majority complementary is acceptable. The feasibility of this approach is based on the fact that different inverter has different sensitivity of voltage. This sensitivity variation is caused by fabrication process variation. Ideally, the delays of inverters will change with a function of voltage $f(v, p)$ that has the same parameters p. When the working voltage changes, all inverters behave the rigorous delay changes based on $f(v, p)$. Then PUF's output will keep stable. However, due to the process variation, parameter p of each inverter varies to each other, causing the irregular delay variations. Furthermore, the scale of additional irregular delay variations under different voltages is much smaller than the scale of inverter's delay. Therefore, the inverter number's inequality mentioned in above cloning scenario cannot realize this reversible feature. If the attacker tries to clone the ROs with the same number of inverters, any specific pair cannot be guaranteed to flip under voltage difference, because the voltage sensitivity is uncontrollable. More than only use delay difference as a one-dimensional constraint, we expand the PUF's security constraint to two dimensions: the delay difference and voltage sensitivity variation. Obviously, under this two-dimensional constraint, cloning is much more infeasible.

Even though there is no theoretical proof shows that the bits generated by ROs will flip under given voltage variation, two features will still guarantee the effectiveness of our dual-voltage anti-cloning approach. 1) With the configurable feature, it is much easier to find the configuration vectors that lead to bit flips under voltage variation. 2) Our approach does not require all the bits to flip. Only a portion of them is sufficient to verify the clone.

VI. Experimental Results

We use our in-house data to validate the uniqueness and reliability of our approach. The configurable RO PUFs are implemented on 9 Xilinx Virtex-5 FPGA boards: 3 ML501's, 3 ML506's, and 3 ML510's. The PUFs are operated at 35°C and 70°C so that we can determine the percentage of flips in PUF-response bit-streams under temperature variation.

Each stage (including an inverter and a multiplexer) of the Configurable RO PUF is implemented by one Look-Up-Table (LUT). The locations of LUTs and the measurement flip-flops are manually specified using Relative Location Constraint so that we can achieve identical placement of all Configurable RO PUFs − this helps increasing the uniqueness and randomness of the PUF's outputs.

A. Uniqueness of Configurable RO PUFs' outputs

Fig. 3 shows the histograms of the inter-chip HD of the outputs of our Configurable OROB PUFs. One special feature of our OROB approach is that we can build even number of inverters in one ring. But we must pick odd number of inverter to form a subRO. In the experiments, we build 15 inverters in every ring. The histogram shows the inter-chip HD under 35°C. The mean HDs and standard deviations of each histogram are 46.44%, 8.23%. The histogram shows that there is not a pair of FPGA boards that produce identical or completely complementary PUF output stream. Even though the number of FPGA boards used in this experiment is small, this result still suggests that our Configurable OROB PUF has near-ideal uniqueness.

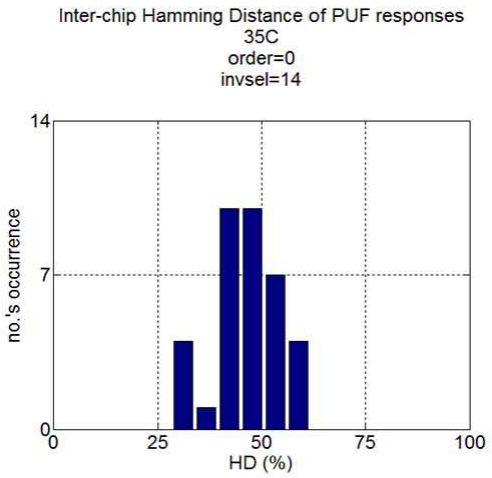

Figure 3. Pairwise Hamming distance (HD) of PUF output bits generated on different chips.

B. Reliability

Using the delay information of inverter collected from 9 FPGA boards at 35°C and 70°C, we construct our traditional RO PUF, 1-out-of-8 RO PUF, neighbor-chained RO PUF, Gao's RO PUF and our configurable OROB PUF, with different lengths and compare their PUF outputs' reliability under temperature variation. For our configurable OROB approach, length n means that we compare $\lfloor n/2 \rfloor$ fastest inverters with $\lfloor n/2 \rfloor$ slowest inverters we assume n is odd. Fig. 4 shows a quantitative comparison of the reliability of our configurable approach and other RO PUF designs. The average percentage of bit flips is calculated by averaging the percentages of flips in PUF response bits of 9 FPGA boards under temperature variation. On average, our configurable

approach always generates more reliable PUF response bits than traditional and neighbor-chained approaches regardless of the length of ring oscillators.

From Fig.4, we can calculate that the average bit flip ratio between OROB and [22]'s approach is $(9.93/22.81 + 7.25/21.37 + 3.01/19.43)/3 = 0.31$. This means compared with [22], our approach reduces the bit flip rate by 69%. The bit flips of all approaches from Fig. 4 are much higher than the experimental results published by the corresponding authors. It is because we implement these PUF at inverter level, and the extra programmable circuit inside FPGA system is very noisy. However, even under such severe testing environment, our OROB approach still achieve very low bit flip rate.

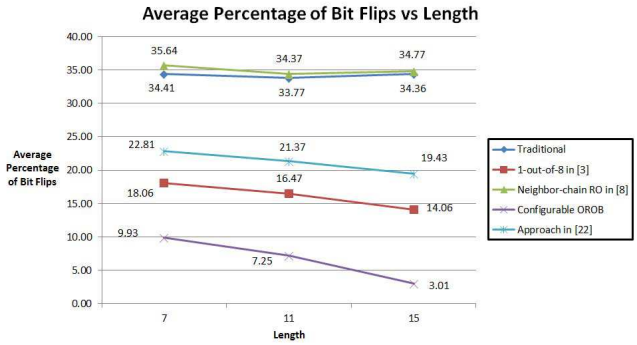

Figure 4. The average Percentage of Bit Flips under different Length

C. Hardware Utilization on FPGA

We define the hardware utilization metric as the number of reliable PUF response bits per one hardware unit. For FPGA implementation, because LUTs are the most basic building block of FPGA architecture, we can consider one LUT as one hardware unit.It might seem that the configurable RO uses more hardware (multiplexers) to implement than the traditional RO, but it is not true in FPGA implementation. Typically, one inverter is implemented in one LUT in other RO approaches. In our approach, we implement an inverter and the multiplexer inside one LUT. So based on the number of response bits generated by 9 FPGA boards for each approach, we can calculate the corresponding necessary number of LUTs. Then we can calculate the FPGA hardware utilization by dividing the total number of reliable bits generated by 9 FPGA boards by the total number of LUTs. Fig. 5 shows our quantitative comparison of the FPGA hardware utilization between our OROB approach and other RO PUF designs. Clearly, our approach has the second best FPGA hardware utilization and is only outperformed by neighbor-chained approached. Because neighbor-chained approach generates many response bits that are dependent due to its re-usage of ring oscillators, its security is much weaker and information entropy is much smaller than our approach. Therefore, our configurable approach has the best trade-off among reliability, hardware cost, and security and stands out as the most attractive candidate for FPGA-based RO PUF.

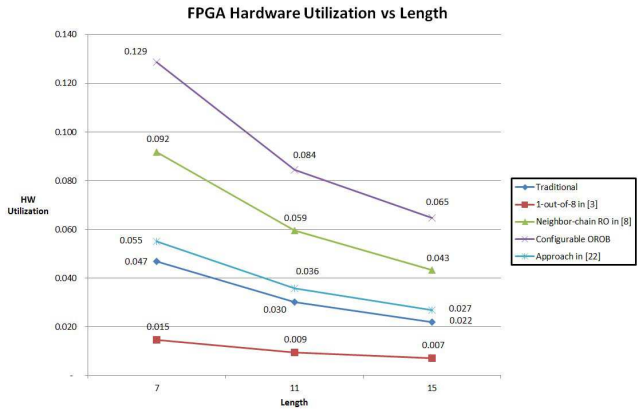

Figure 5. Hardware Utilization Comparison in FPGA

The hardware utilization of our OROB approach is $(0.129/0.055 + 0.084/0.036 + 0.065/0.027)/3 = 2.36$ times higher than [22]'s approach, which means our approach improve the hardware utilization by 136%.

The major reason of the high hardware utilization of our approach is that we break one RO into two subRO. To generate response bits, we need two response cycles, because two subROs cannot be configured at same time. However, since the major problem of PUF is to solve the secure issues, compared with PUFs reliability and security, the cost of longer response time is a minor issue.

D. Dual Voltage based Anti-Cloning Approach

Because our in-house data doesn't include the voltage variation conditions, so we use the Virginia Tech (VT)'s public PUF data [20] to validate this anti-cloning approach. Although the data consists of only frequencies in RO level, we can build each RO composed of an odd number of small ROs. The frequencies of small ROs are come from the VTs data. Because we use the reverse pattern to detect a physically cloned PUF, we want the pattern to be reliable so we define a threshold R_{th}. If the delay difference between two subROs is bigger than R_{th} at both V_1 and V_2 and the responses at V_1 and V_2 are different, we will consider it as a reliable flip and use it for anti-cloning; otherwise we cannot use it.

Table 1 shows the flipping patterns for five XC3S500E FPGA boards from VT data. The 2nd columns are the PUF secret bits in hexadecimal generated at normal supply voltage 1.2V. The 3rd column shows the PUF secret bits under 1.44V. If we convert the hexadecimal data into binary,

Chip ID	PUF data at 1.2V	PUF data at 1.44V	Flipping rate (%)
D059546	59F4A484	A2585843	68.75
D113702	2DE8523E	C789604E	43.75
D113938	D2225B59	D87ADBDD	25.00
D225158	182C25ED	C07CA419	40.63
D225159	48C85CC5	BA07FA93	59.38

Table I
FLIPPING PATTERN

we can clearly see that the data in 3rd column is much different with the data in 2nd. The 4th column is computed by dividing the number of bit flips at 1.44V by the number of secret bits. The average flip rate of these 5 chips is 47.5%. This means the testing voltage will generate the bitstream with 47.5% HD compared with the normal working voltage. This makes cloning very difficult. A straightforward way to detect a clone is following: (1) set the voltage to 1.2V and get the PUF secret bits R_1; (2) set the voltage to 1.44V and get the PUF response bits R_2, and finally; (3) check whether some certain bits in R_2 will flip compared with R_1. If yes, the PUF is genuine; otherwise it is a physical clone. This result shows the feasibility of finding the configuration vectors that yields a high flipping rate to detect a physically cloned PUF.

CONCLUSION

In this paper, we propose a novel PUF schematic that configures the RO after fabrication based on the real inverter delay measurement. Compared to existing RO PUF designs, our method generates more PUF secrets bits which are more reliable in the presence of environment variation. Also our approach enhances the RO PUF's resistance against existing physical cloning attack by utilizing the inverters' voltage sensitivity. All of our proposed approaches have been validated by experiments on our in-house data and public RO PUF data.

ACKNOWLEDGEMENT

This project was supported in part by AFOSR MURI under award number FA9550-14-1-0351 and an ARO special program award.

REFERENCES

[1] U. Ruhrmair, X. Xu, J. Solter, A. Mahmoud, M. Majzoobi, F. Koushanfar, and W. Burleson. "Efficient Power and Timing Side Channels for Physical Unclonable Functions". In *Proc. International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, pp. 476-492, 2014.

[2] J. Guajardo, S. S. Kumar and P. Tuyls. "FPGA Intrinsic PUFs and their Use for IP Protection". In *Proc. International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, pp.63-80, Sep. 2007.

[3] G. E. Suh and S. Devadas. "Physical Unclonable Functions for Device Authentication and Secret Key Generation". In *Proc. 44th ACM/IEEE Design Automation Conference (DAC)*, pp. 9-14, Jun. 2007.

[4] R. Maes G.J. Schrijen S. Kumar, J. Guajardo and P. Tuyls. "The Butterfly PUF: Protecting IP on every FPGA". In *Proc. IEEE International Workshop on Hardware Oriented Security and Trust (HOST)*, pp.67-70, Jun. 2008.

[5] W. B. D. Holcomb and K. Fu, "Initial sram state as a fingerprint and source of true random numbers for rfid tags," In *Proc. Conference on RFID Security*, Jul. 2007.

[6] A. Maiti and P. Schaumont. "Improving the Quality of a Physical Unclonable Function Using Configurable Ring Oscillators". In *Proc. IEEE International Conference on Field Programmable Logic and Applications (FPL)*, pp. 703-707, Sep. 2009.

[7] C. Yin and G. Qu. "Temperature-Aware Cooperative Ring Oscillator PUF," In *Proc. IEEE International Workshop on Hardware-Oriented Security and Trust (HOST)*, pp. 36-42, July 2009.

[8] A. Maiti and P. Schaumont. "Improved Ring Oscillator PUF An FPGA-friendly Secure Primitive". *Journal of Cryptology*, vol. 24, no. 2, pp. 375-397, 2011.

[9] D.Karakoyunlu, B. Sunar, "Differential Template Attacks on PUF Enabled Cryptographic Devices". In *Proc. IEEE International Workshop on Information Forensics and Security (WIFS)*. pp. 16, Dec 2010.

[10] D. Merli, D. Schuster, F. Stumpf, G. Sigl, "Side-Channel Analysis of PUFs and Fuzzy Extractors". In *TRUST*, pp. 3347, 2011.

[11] D. Merli, D. Schuster, F. Stumpf, and G. Sigl, "Semi-invasive EM attack on FPGA RO PUFs and countermeasures," in *Workshop on Embedded Systems Security (WESS)*, Oct. 2011.

[12] D. Merli, J. Heyszl, B. Heinz, D. Schuster, F. Stumpf, G. Sigl, "Localized electromagnetic analysis of RO PUFs". In *HOST*, pp.19-24, June 2013.

[13] C. Helfmeier; C. Boit; D. Nedospasov; J.-P. Seifert, "Cloning Physically Unclonable Functions". In *HOST*, pp.1-6, June 2013.

[14] U. Rhrmair, F. Sehnke and J. Schmidhuber, "Modeling attacks on physical unclonable functions," In *Proc. ACM Computer and Communication Security Conference (CCS)*, Oct. 2010.

[15] A. Mahmoud, U. Rhrmair, M. Majzoobi, F. Koushanfar. "Combined Modeling and Side Channel Attacks on Strong PUFs". *IACR Cryptology ePrint Archive* , 2013.

[16] X. Xin, J. Kaps, K. Gaj, "A Configurable Ring-Oscillator-Based PUF for Xilinx FPGAs,"In *Euromicro Conference on Digital System Design (DSD)*, pp.651-657, 2011.

[17] M. Majzoobi, E. Dyer, A. Elnably. F. Koushanfar, "Rapid FPGA delay characterization using clock synthesis and sparse sampling," In *IEEE International Test Conference (ITC)*, pp.1-10, Nov. 2010.

[18] D. Forte, A. Srivastava, "On improving the uniqueness of silicon-based physically unclonable functions via Optical Proximity Correction," In *ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp.96-105, June 2012.

[19] R. Kumar, H.K. Chandrikakutty, S. Kundu, "On improving reliability of delay based Physically Unclonable Functions under temperature variations," In *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pp.142-147, June 2011.

[20] A. Maiti, P. Schaumont, "Research on Physical Unclonble Functions (PUFs) at SES Lab, VT," http://rijndael.ece.vt.edu/puf/main.html.

[21] D. Lim , J. W. Lee, B. Gassport, et al., "Extracting secret keys from integrated circuits," *IEEE Trans. VLSI*, vol. 13, no. 10, pp. 1200-1205, 2005.

[22] M. Gao, K. Lai, G. Qu, "A Highly Flexible Ring Oscillator PUF," In *Proc. 51th ACM/IEEE Design Automation Conference (DAC)*, pp.1-6, 2014.

[23] J. Zhang, G. Qu, Y. Lyu, and Q. Zhou. "A Survey on Silicon PUFs and Recent Advances in Ring Oscillator PUFs". *J. Comput. Sci. Technol.*, 2014, 29, 4, pp. 664-678.

[24] J. Zhang, Y. Lin, Y. Lyu, and G. Qu, "A PUF-FSM Binding Scheme for FPGA IP Protection and Pay-per-Device Licensing". *IEEE Trans. Inf. Forensics Secur.*, vol. 10, no.6, pp. 1137-1150, 2015.

A Machine Learning Attack Resistant Dual-mode PUF

Qian Wang, Mingze Gao, Gang Qu
Department of Electrical and Computer Engineering and Institute of Systems Research
University of Maryland, College Park, MD
{qwang126,mgao1,gangqu}@umd.edu

ABSTRACT

Silicon Physical Unclonable Function (PUF) is arguably the most promising hardware security primitive. In particular, PUFs that are capable of generating a large amount of challenge response pairs (CRPs) can be used in many security applications. However, these CRPs can also be exploited by machine learning attacks to model the PUF and predict its response. In this paper, we first show that, based on data in the public domain, two popular PUFs that can generate CRPs (i.e., arbiter PUF and reconfigurable ring oscillator (RO) PUF) can be broken by simple logistic regression (LR) attack with about 99% accuracy. We then propose a feedback structure to XOR the PUF response with the challenge and challenge the PUF again to generate the response. Results show that this successfully reduces LR's learning accuracy to the lower 50%, but artificial neural network (ANN) learning attack still has an 80% success rate. Therefore, we propose a configurable ring oscillator based dual-mode PUF which works with both odd number of inverters (like the reconfigurable RO PUF) and even number of inverters (like a bistable ring (BR) PUF). Since currently there are no known attacks that can model both RO PUF and BR PUF, the dual-mode PUF will be resistant to modeling attacks as long as we can hide its working mode from the attackers, which we achieve with two practical methods. Finally, we implement the proposed dual-mode PUF on Nexys 4 FPGA boards and collect real measurement to show that it reduces the learning accuracy of LR and ANN to the mid-50% and low 60%, respectively. In addition, it meets the PUF requirements of uniqueness, randomness, and robustness.

KEYWORDS

Physical Unclonable Functions; Configurable ring oscillator; Modeling attacks; Artificial Neural Network

ACM Reference Format:
Qian Wang, Mingze Gao, Gang Qu. 2018. A Machine Learning Attack Resistant Dual-mode PUF. In *GLSVLSI '18: 2018 Great Lakes Symposium on VLSI, May 23–25, 2018, Chicago, IL, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3194554.3194590

1 INTRODUCTION

Physical Unclonable Functions (PUFs) have been widely accepted as a promising hardware security primitive and recently adopted in many security applications such as hardware fingerprinting and authentication, secure key generation, and secure storage for ciphers. One of the most important features of PUF is that it can map challenges (inputs) to responses (outputs) based on the intrinsic physical variation in the hardware devices. The properties of unclonability and unpredictability among others give PUFs certain advantages over conventional cryptography based solutions in many security applications, in particular on systems where computing, power, and storage resources are limited.

There are three representative PUFs that can generate challenge-response pairs (CRPs) exponential to the number of bits in the challenges: the classic arbiter PUFs [1] where the challenge determines the delay paths, the reconfigurable ring oscillator (RO) PUFs [2] where the challenge determines whether the inverter in each stage will be included in the delay path, and the bistable ring PUF [3] which can generate a stable random bit based on the challenge applied to each of its stage. However, due to the fact that the exponential number of CRPs in these PUFs are generated from hardware units (such as MUXes and inverters) that are linear to the number of bits in the challenge, it is not surprising that they are facing severe threats from (machine) learning based attacks. In such modeling attacks, the attackers can utilize a subset of CRPs to build mathematical models for the PUF and then use the model to predict the response of the other challenges thus breaking the PUF. Arbiter PUFs and reconfigurable RO PUFs are vulnerable to linear learning techniques such as logistic regression (LR) [4–6] and the bistable ring PUF can be broken by differential and linear analysis [7].

Our goal is to seek novel PUF structures that are highly resistant to machine learning attacks based on both linear models and non-linear models. This paper makes the following contributions towards achieving this goal:

(1) We apply the machine learning attacks on conventional arbiter PUF and RO PUF. Both the linear model based attack (Logistic regression, LR) and the non-linear model based attack (artificial neural network, ANN) can break the PUFs with about 99% accuracy.
(2) We introduce a feedback structure in the PUF where we perform bitwise XOR of the challenge and its response and use the result as input to challenge the PUF again in order to create the real response. This hides the direct relationship between challenge and real response because the attacker cannot observe the first response. This approach successfully reduces LR's learning ability to the low 50%, but ANN's learning accuracy is still above the 80%.
(3) We propose a novel dual-mode feedback PUF on the reconfigurable RO-PUF platform which can behave as either an RO PUF or a bistable ring PUF to confuse the attacker. We implement the dual-mode PUF on FPGA and the measured

GLSVLSI '18, May 23–25, 2018, Chicago, IL, USA

ACM ISBN 978-1-4503-5724-1/18/05…$15.00
https://doi.org/10.1145/3194554.3194590

data show that we are able to reduce the learning accuracy of LR to the mid-50% and that of the ANN to the low 60%.

(4) We evaluate the quality of the proposed dual-mode PUF in terms of randomness, uniqueness, and reliability (against temperature changes). The results based on real FPGA chip measurement show that the dual-mode PUF meets all these requirements.

The remainder of the paper is organized as follows: Section 2 introduces the related PUF designs and their vulnerability to modeling attacks; Section 3 presents the proposed design of configurable dual-mode PUF. In addition, the challenge obfuscation and masking schemes are introduced in this section; Section 4 provides the experimental results for the modeling attack and demonstrates that the dual-mode PUF could resist the state-of-art modeling attacks. Section 5 presents the quality analysis of the innovated PUF structure, and then Section 6 concludes this paper.

2 PUF DESIGNS AND THEIR VULNERABILITY TO MODELING ATTACKS

For machine learning attacks, it is necessary to collect reasonably amount of training data. Therefore, in this section, we briefly discuss PUFs capable of generating large amount of CRPs and their vulnerability to modeling attacks.

2.1 Arbiter PUF and basics of modeling attack

The classic arbiter PUF [1] consists of two MUX arrays, at each stage of the array there are two MUXes, both connecting to the two MUXes in the next stage. Two signals start from the two MUXes in the first stage simultaneously and choose their path according to the value of the selection bit to the MUX in each stage. These selection bits are known as challenge. Due to manufacture variation, the two signal will not go through the last stage at the same time and this delay discrepancy defines a PUF bit, which is called response.

In a modeling attack, the input of the training data is the challenge of PUF: $C = c_1 \cdots c_k$; and the label will be the 1-bit response $r \in \{0, 1\}$. The arbiter PUF could be modeled as a linear additive model since a response bit is generated based on the summation of delay segment in each stage depending on the challenge C[4, 5]. The delay difference Δ can be expressed as

$$\Delta = \omega^T \Phi \tag{1}$$

where ω is the delay vector for each of the segment in the arbiter PUF structure and Φ is the following function of the k-bit challenge C

$$\Phi(C) = (\Phi^1(C), ..., \Phi^k(C), 1)^T \tag{2}$$

where $\Phi^j(C) = \prod_{i=j}^{k}(1 - 2c_i)$ for j = 1, ..., k (Note: different types of PUF would have different Φ function to map a challenge C to a real value). If $\Delta > 0$, the response bit r will be '1'. Otherwise, r will be '0'.

The goal of a machine learning attacker is to find out an estimate of ω that can represent the actual delay vector of physical PUF structure. With sufficient number of CRPs, the delay information ω of each stage in the arbiter PUF can be learned with high accuracy [4, 8, 9]. Now the general assumption is that such attack requires less than 10% of the total number of CRPs [10].

There are several methods proposed to resist the modeling attacks on PUFs. One solution is to complicate the modeling attack by altering the PUF structure and reducing the correlation between challenges and responses, e.g., the XOR-mixed PUF [5] and the twisted PUF [11]. These attempts provide some modest defenses over the linear learning algorithm; however, simply adding non-linearity in the PUF structure cannot defend more powerful learning techniques that could solve non-linear classifications.

Another efficient approach to confuse the adversary is to introduce redundancy in the challenges[12]. In this approach, the challenges are obfuscated either by adding another redundant challenge or by padding some unused bits in the original challenge in order to hide the real challenge from the adversary. This will definitely increase hardware cost in both the PUF instance and the authentication server. Also, it will cause longer response time as both the PUF and the server need a pre-processing procedure before the authentication.

2.2 Reconfigurable RO PUF and Bistable PUF

The original RO PUF uses the delay difference between two ring oscillators with an odd number of inverters to create one bit of information. Its main application is to generate a secret key and does not support CRPs. Modification to the RO PUF structure has been proposed to enable CRP, for example, by having two inverters at each stage and use the challenge bit to select one. In the highly flexible reconfigurable RO PUF, a selection bit decides at each stage whether the inverter will be included in the RO. Thus different challenges, which are called configuration vectors, may result in different response [2, 13]. Delay in the reconfigurable RO PUF structure could be mapped as linear additive functions, which demonstrate vulnerabilities to machine learning methods.

The concept of bistable ring PUF (BR PUF) was proposed in [3] where even number of inverters are connected from head to tail to force it into two arbitrary Boolean states, logic 0 or logic 1. The security evaluation of BR PUF has been studied in [11], where the resilience of BR PUF against modeling attacks is evaluated and it is stated that a possible mapping relationship between challenge and response may cause the PUF response to be predictable. Furthermore, it shows that the alternative twisted bistable rings PUF (TBR PUF) is immune to learning algorithms. However, it still leaks some correlations between the challenges and the final stable states, which could be learned by specific learning strategy.

Our proposed dual-mode PUF design is developed based on the architecture of the reconfigurable RO PUF but allow the use of an even number of inverters to provide features of bistable ring PUF. Thus, depending on the challenge (i.e. the configuration vector), the dual-mode PUF may work either as an RO PUF or as a BR PUF. Because these are two different types of PUF and existing machine learning models to attack them are different (to the best of our knowledge, there has not been any efficient model reported working for both the delay ring and bistable ring), it will be very challenging for model-based attacks to break the proposed dual-mode PUF.

3 DESIGN OF DUAL-MODE PUF

In the following, we will first discuss the design of unit cell and propose our dual mode topology. Then we will introduce the technique applied in the dual mode PUF design to thwart the modeling attacks.

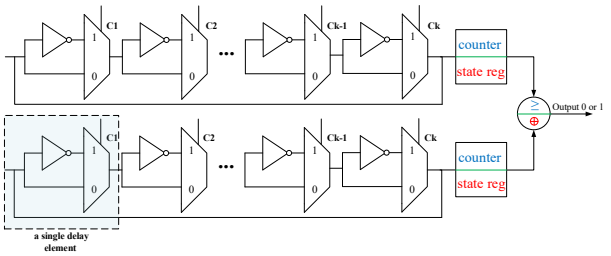

Figure 1: The architecture of K-stage reconfigurable dual-mode PUF, a single delay element is marked in shaded box

3.1 Dual-mode PUF structure

As shown in Fig.1, the basic delay element which consists of an inverter and a multiplexer has been placed in parallel and connected from head to tail. The corresponding challenge bit of stage i is c_i which controls the inverter to be selected or not. If the selection bit is "1", the delay of this stage would be $d_{path1} = d_{mux} + d_{inv} + d_{wire}$; otherwise, the delay of the inverter would not be included and the delay would be $d_{path0} = d_{mux} + d_{wire}$.

By connecting the basic delay element, we build the dual mode configurable PUF. Unlike those delay based RO-PUFs, which require odd number inverters in the chain to guarantee the oscillation, our new design would work on both odd and even number inverters. More precisely, the working mode of our PUF depends on the parity of the challenge as shown in Fig.1. For the odd mode, the circuit drawn in a rectangular box works as a counter which counts the oscillating frequency, and the circular shape module represents a comparator which compares the frequency and generates 1-bit response based on the difference. For the even mode, the functionality of the rectangular box is a state register which stores the stable state is whether 5 (0101) or A (1010). The following module is an exclusive-or function generates 1-bit response from two separated rings to make the even structure compatible to the odd structure. Moreover, the exclusive-or (XOR) logic has the linearly non-separable attribute, which would offering intrinsic resilience to linear machine learning attacks. In addition, the output of XOR logic is of uniform distribution as long as one of the inputs is uniformly distributed. Therefore, XOR logic gate standouts other logics like AND, OR, etc. in our design.

3.2 Overcoming the learning vulnerability of bistable ring

Our basic delay element shown in Fig.1 has some significant changes compared to the bistable ring shown in Fig.2. First, we only deploy one inverter and the compared delay path as $path_0$ is generated by the intrinsic delay of the wire and the MUX. Compared to the bistable ring, we get rid of the duplicated inverter and the demultiplexer for each stage which could definitely save the hardware cost.

Besides, our design overcomes the modeling attack vulnerability of bi-stable ring discussed in [7]. In the traditional bistable ring, the driving behavior of one fixed inverter is compared to the inverter in the adjacent stage. With this assumption, the author defined the difference between the pull-up and pull-down capability of the inverter as strength t_i for the ith stage. And all the even stages will contribute toward the positive response as t_i, the odd stages will contribute toward the strength as $-t_i$. Thus the attacker can model the competition behavior based on the strength of each stage.

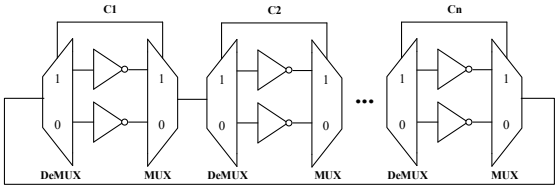

Figure 2: Delay element of bistable ring PUF

However, this model would not work in our dual mode design because we do not fix the odd stages to be pull-up and the even stages to be pull-down. On the contrary, each stage in our design could either be the pull-up or pull-down determined by the input challenges. When the selection bit for this stage is 0, the inverter in this stage would be bypassed and the input signal would directly drive on the next selected inverter. This could avoid the fixed adjacent "strong-weak" pattern in the bistable ring. Moreover, the connecting wires and the MUXs could influence on the output resistance of the inverter which indeed impacts the driving capability of it. All of the aspects make it infeasible to accurately model the behavior of our new design of bistable ring by the pull-up and pull-down strength.

3.3 Hiding the working mode by challenge obfuscation and masking

The work mode of the dual-mode PUF is purely determined by the parity of the input challenge. If the adversary could learn the parity of challenge in advance, he may build two separate models for the dual-mode PUF and the complexity of learning the PUF would be halved. Here, we designed two mechanisms to hide the working mode of our dual mode PUF to mitigate the security concern as the adversary could know the parity of the challenges.

3.3.1 Challenge obfuscation. Fig. 3 shows the flow chart of our challenge obfuscating and masking approaches. The original challenge C would pass the challenge obfuscated circuit first, and several bits of C are modified to make the parity of the new generated challenge C' is unknown to the adversary. As a result of this, the attacker cannot build separate learning models even though he knows how the dual mode PUF works. Note that the challenge obfuscation circuit is built with the PUF instance in hardware; there is no I/O ports for the adversary to acquire the new generated challenge C'.

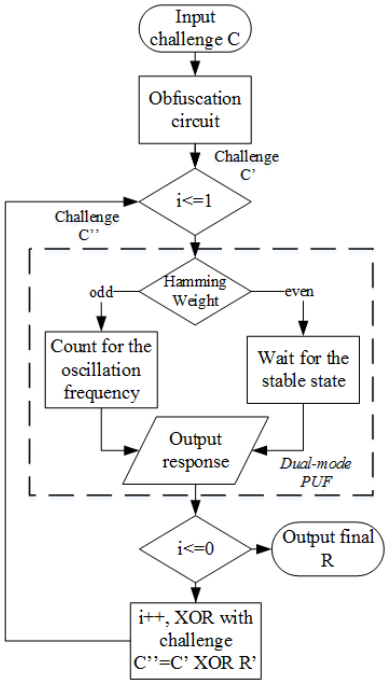

Figure 3: The flow chart of two-round Dual mode PUF illustrating the challenge obfuscation and masking

A 16-bit challenge case is shown in Fig.4. The input challenge is divided into 4 parts (the length of the challenge is the multiple of 4). We determinatively choose one bit from each part (colored in yellow), and generate the boolean value by bitwise AND or OR operation. Next, we assign the new boolean value to the selected positions (e.g. bit 8 and bit 9 in this example). The goal for this challenge obfuscation is to confuse the adversary on the parity of the challenge. Even though some bits are unchanged from the original challenge after the obfuscation, this would not hamper the effect we expect. We conduct statistical analysis on our obfuscation mechanism and results show that for both even and odd inputs, the parity flip rate is 0.5 which is as the same as random flip rate.

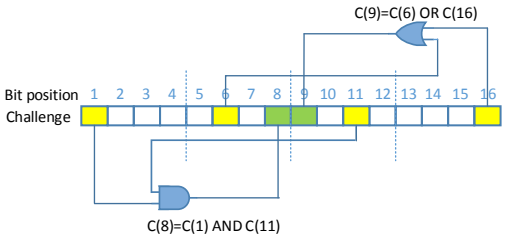

Figure 4: Challenge obfuscation example, when the challenge bit is 16 and each sub-block has 4 bits

3.3.2 Challenge masking. From the first step, the obfuscation circuit successfully hides the parity of the input challenge but some bits of are still exposed to the attacker. It is necessary to find an approach to hide the input pattern from the malicious attackers without hampering the PUF property. Thus, we design a two-round masking scheme to conceal the input challenge to the adversary shown in Fig. 3. In the first round, the challenge after the obfuscation circuit C' would input to the PUF directly and the same length output is generated as $R' = \text{PUF}(C')$, where we place the same number of rings to keep the length of output unchanged. Then, the response R' would be exclusive-or with the input challenge C' in obtaining a new challenge as $C'' = C' \oplus R'$ which is the input to the PUF in the second round. This procedure masks the challenge of the second round which not only conceals the parity but also hides the relationship of challenge and response to the attacker. The intermediate result R' and C' would input to the PUF in the next step immediately, thus leaving no time for the attacker to intercept them. Therefore, the adversary can only use the input challenge with two-round response (C, R) to train a model. The direct additive relationships between challenge and response pairs are confused by the XOR function and the two-round iteration. The intermediate responses are required in order to derive the final response, but the intermediate results are invisible. Hence, those CRPs leak limited information to an adversary in modeling the PUF.

After implementing the challenge obfuscation and masking, the adversary faces a big challenge to find out the appropriate model for the PUF. Since the parity of the input challenges is hidden to the adversary, he can only randomly guess the combination of work mode of the PUF from 4 possible patterns: odd-odd, odd-even, even-odd and even-even. However, the adversary has no knowledge on how to distinguish the CRPs from the four patterns. If he arbitrary choose one mode, there are 3/4 noise CRPs which he could not figure out. We will theoretically prove the upper bound of the prediction of accuracy in the following. It has been demonstrated that a relationship between machine learning complexity and noise of input labels is subject to the noise bit corruption rate η , where a value of $\eta = 0$ means that none of the labels are corrupted, and $\eta = 0.5$ means that values are completely corrupted by random noise.

The adversary can guess, among all the candidate mode, given an incoming CRP, the adversary can have 1/D chance to guess the correct working mode. In our case, for two-round dual mode PUF, as D=2, the successful rate of the guess would be $1/D \times 1/D = 1/4$. We define the mode uncertainty for one step would be as $\varepsilon_u(D) = 1 - 1/D$. Thus, for the dual mode PUF the mode uncertainty would be 1-1/4=3/4. Besides, we assume that the learning noise for a certain mode would be η_e , and the learning noise after apply the working mode uncertainty would be

$$\eta_a(n) = 1 - [\varepsilon_u(D) \cdot \eta_e(n) + (1 - \varepsilon_u(D)) \cdot (1 - \eta_e(n))] \quad (3)$$

We can get the noise of input labels by experiment and plug into the equation to get the bound for the final noise level. By plugging the noise level of 0.046 from experiment [14], the noise after two rounds would be a level of 0.273, and the advantages would grows with D.

4 EXPERIMENTAL RESULTS

We choose the most representative two methods (LR and ANN) to test our dual-mode PUF design. To make a comparison, we also

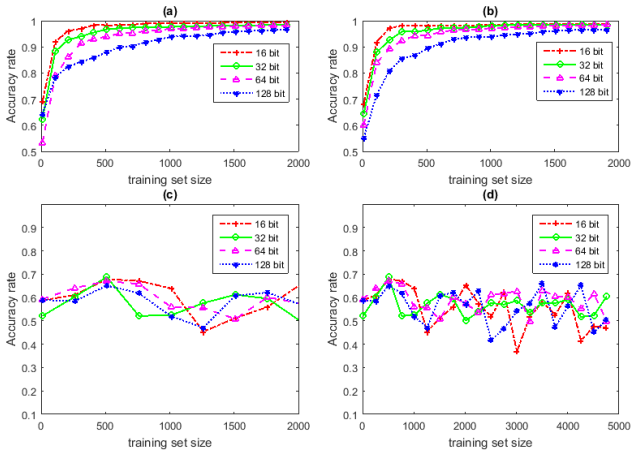

Figure 5: Results of modeling attack (logistic regression) on (a) RO PUF, (b) Arbiter PUF, and (c) Dual mode PUF with 2000 CRPs (d) Dual mode PUF with 5000 CRPs

test the traditional arbiter PUF and RO-PUF by the same learning algorithms. Also, we will verify the potential of anti-modeling on different proposed dual-mode PUF structures.

4.1 Results from logistic regression

4.1.1 Test results on a single pair of ROs. We first test a single pair of RO with different stage size as 16, 32, 64, 128. The CRPs for arbiter and RO-PUF are from the simulated data. Paper [5] demonstrates that modeling attacks would work both on simulated and silicon data, and the only difference is that the results on simulated data are noise free. However, by using more CRPs in training stage, results from the real silicon could achieve the same accuracy rate (e.g., 99%) compare to the simulated data.

The logistic regression attack results for three kinds of PUFs are shown in Fig.5. We sweep the training size from 10 to 2000 for the traditional arbiter and RO-PUFs and enlarge it to 5000 for our dual-mode PUF. The y-axis represents the learning accuracy rate and the maximum value is 1 which means the PUF responses can be completely predicted. For ideal learning resistant case, the accuracy should be around 0.5, meaning that the machine learning prediction is no better than the random guess.

It can be seen in Fig.5 (a) and (b) the learning rates are near 1 (100%) when the training set is around 2000 which indicates that the logistic regression successfully models the arbiter and RO-PUF. The learning accuracy got from the dual mode PUF is dramatically different from the above, shown in the 5 (c). The learning rate is oscillating around 0.55 nearing the randomly guess rate. Moreover, even we enlarge the training set to 5000 shown in the Fig. 5 (d)(nearly double size we used for delay based PUFs), the predication rates has no trend in increasing. From the results summarized above, it is confident to conclude that our dual mode PUF could resist the logistic regression learning attack.

4.1.2 Test results on PUF array. Further, we apply the attack method on PUF array structure in section 3.3. As discussed before, the attacker may group the challenge based on its parity and build different learning models separately. To solve this problem, we propose the array structure to break the linear relationship of the challenge and response as discussed in section 3.3. We place in total 32 ring instances on the FPGA and each ring consists 32 stages. We name this structure as (32,32) PUF array. To keep consistent, the single pair of ROs we studied above is named as (32,1) structure.

The array structure masks the actual challenge of the second round C'' by the first round response R' as $C'' = C' \oplus R'$. We choose XOR function because it could add resilience to the linear learning model. As a result of this, the learning accuracy rate shown in Table 1 goes down to around 0.5 when using logistic regression to model the masked PUF arrays. To make a comparison, we also list the attack result of a single pair of 32-bit ROs shown in the table as (32,1). Results from the Table 1 shows that the logistic regression method fails in modeling the array structure. The reason is because LR is a linear learning algorithm and only works for the linear separable models. The masking method applied on the input challenges introduces the intrinsic non-linearity, thus making the logistic regression ineffective. However, the state-of-art neural network could overcome the limitation of the linear learning algorithms. Therefore, we also test the same array structure design under the neural network based learning method.

4.2 Results from neural network

The ANN method could model non-linearity based on the multilayer network structure and the non-linear activation function. The ANN learning network structure we used in the experiment is a 2-layer MLP with 40 nodes in each layer. The learning results from the ANN method are shown on the right side of Table 1. We observe that the ANN method greatly improves the learning rate for the arbiter PUF and RO PUF, but still fails to learn the dual-mode PUF.

Table 1: Attack results for three type of PUFs

Type	LR		ANN	
	(32,32)	(32,1)	(32,32)	(32,1)
Arbiter PUF	0.5132	0.9851	0.8192	0.9933
RO PUF	0.5009	0.9906	0.8132	0.9982
Dual Mode PUF	0.5404	0.5580	0.6162	0.6105

5 PUF QUALITY ANALYSIS

In this section, we evaluate the quality of the two kind of implementations for the dual-mode PUF: one is the single pair of ROs which has 16 bit, 32 bit, 64 bit and 128 bit challenge width respectively; the other is the array structure, where n pairs of dual-mode rings (n delay elements for each) are placed and the length of the controlled configurable bit is chosen as $n = 16, 32$.

We evaluate the uniformity(randomness of the output) of a single pair RO based on the response collected from FPGA board. The uniformity checks the distribution of "0" and "1" under different configuration inputs. The good PUF should have the ideal value of 0.5 as uniformly distributed. The results of uniformity per bit

under 50000 random selected challenges are shown in the table 2. We further examine the same statistical properties of the (16,16) and (32,32) PUF arrays. Results show that conducting XOR on the intermediate response will have extra good effects on the uniformity. This is because the output of XOR gate would have uniform distribution if the input is uniformly distributed. From the results in Table 2, we could observe that the (32,32) array has better properties in uniformity, which makes it as a good candidate in real applications. The slight bias from the ideal value (0.5) might be caused by multiple reasons. First, since we use the data acquired from the FPGA board, factors such as the disturbance from FPGA surrounding logics and the noise could drift the response, this phenomenon has also been observed in the FPGA based PUFs [15]. As lots of effective solutions have been discussed in the literature to solve this [16], this problem is not the key point we should follow in our paper.

Table 2: Statistical evaluation on the dual mode PUF with 50K CRPs

Type	**16**	**32**	**64**	**128**	**(16,16)**	**(32,32)**
Uniformity	0.4108	0.4465	0.4874	0.4923	0.4377	0.4765

To analyze the reliability of our dual-mode PUFs, we also test the 32-bit PUF instance at several different temperatures from 0°C to 65°C. Taking the CRPs measured at 25°C as the reference, we calculated the average bit flip rates under each temperature. Fig 6 shows the results got from different temperatures, which confirms the temperature sensitivity observed in others' implementations [3]. However, the temperature sensitivity of PUFs has already been solved either by error correction [17] or by the extended authentication protocol for environmental-condition-sensitive PUFs[18]. We could take use of those techniques to overcome the reliability drawback of our dual-mode PUF.

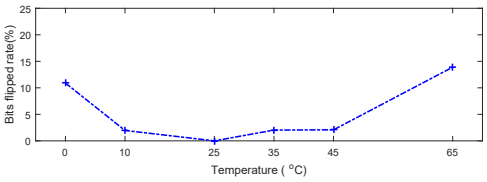

Figure 6: Reliability of dual mode PUF

6 CONCLUSION

Modeling attack is one of the most powerful threats to the PUF based security applications. In this paper, we propose such a PUF design based on both the delay-based PUF and the memory-based PUF which could work effectively in defending the modeling attacks. We implemented the PUF design on the Nexys 4 FPGA board and collected real challenge-response pairs from the board. We tested the learning results using those data and it demonstrated that our design could successfully resist the modeling attacks.

ACKNOWLEDGMENT

This work was supported in part by AFOSR MURI under award number FA9550-14-1-0351.

REFERENCES

[1] Blaise Gassend, Dwaine Clarke, Marten Van Dijk, and Srinivas Devadas. Silicon physical random functions. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 148–160. ACM, 2002.

[2] Mingze Gao, Khai Lai, and Gang Qu. A highly flexible ring oscillator puf. In *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2014.

[3] Qingqing Chen, György Csaba, Paolo Lugli, Ulf Schlichtmann, and Ulrich Rührmair. The bistable ring puf: A new architecture for strong physical unclonable functions. In *Hardware-Oriented Security and Trust (HOST), 2011 IEEE International Symposium on*, pages 134–141. IEEE, 2011.

[4] Ulrich Rührmair, Frank Sehnke, Jan Sölter, Gideon Dror, Srinivas Devadas, and Jürgen Schmidhuber. Modeling attacks on physical unclonable functions. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 237–249. ACM, 2010.

[5] Ulrich Rührmair, Jan Sölter, Frank Sehnke, Xiaolin Xu, Ahmed Mahmoud, Vera Stoyanova, Gideon Dror, Jürgen Schmidhuber, Wayne Burleson, and Srinivas Devadas. Puf modeling attacks on simulated and silicon data. *IEEE Transactions on Information Forensics and Security*, 8(11):1876–1891, 2013.

[6] Qian Wang, An Wang, Gang Qu, and Guoshuang Zhang. New methods of template attack based on fault sensitivity analysis. *IEEE Transactions on Multi-Scale Computing Systems*, 3(2):113–123, 2017.

[7] Dai Yamamoto, Masahiko Takenaka, Kazuo Sakiyama, and Naoya Torii. Security evaluation of bistable ring pufs on fpgas using differential and linear analysis. In *Computer Science and Information Systems (FedCSIS), 2014 Federated Conference on*, pages 911–918. IEEE, 2014.

[8] Yuntao Liu, Yang Xie, Chongxi Bao, and Ankur Srivastava. An optimization-theoretic approach for attacking physical unclonable functions. In *Proceedings of the 35th International Conference on Computer-Aided Design*, page 45. ACM, 2016.

[9] Yuntao Liu, Yang Xie, Chongxi Bao, and Ankur Srivastava. A combined optimization-theoretic and side-channel approach for attacking strong physical unclonable functions. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 26(1):73–81, 2018.

[10] Meng-Day Mandel Yu, Matthias Hiller, Jeroen Delvaux, Richard Sowell, Srinivas Devadas, and Ingrid Verbauwhede. A lockdown technique to prevent machine learning on pufs for lightweight authentication. *IEEE Transactions on Multi-Scale Computing Systems*, 2(3):146–159, 2016.

[11] Xiaolin Xu, Ulrich Rührmair, Daniel E Holcomb, and Wayne Burleson. Security evaluation and enhancement of bistable ring pufs. In *International Workshop on Radio Frequency Identification: Security and Privacy Issues*, pages 3–16. Springer, 2015.

[12] Yansong Gao, Damith C Ranasinghe, Said F Al-Sarawi, Omid Kavehei, and Derek Abbott. Emerging physical unclonable functions with nanotechnology. *IEEE access*, 4:61–80, 2016.

[13] Mingze Gao, Khai Lai, Jiliang Zhang, Gang Qu, Aijiao Cui, and Qiang Zhou. Reliable and anti-cloning pufs based on configurable ring oscillators. In *Computer-Aided Design and Computer Graphics (CAD/Graphics), 2015 14th International Conference on*, pages 194–201. IEEE, 2015.

[14] Meng-Day Yu, Ingrid Verbauwhede, Srinivas Devadas, and David M'RaïĹhi. A noise bifurcation architecture for linear additive physical functions. In *Hardware-Oriented Security and Trust (HOST), 2014 IEEE International Symposium on*, pages 124–129. IEEE, 2014.

[15] Daniel E Holcomb, Wayne P Burleson, and Kevin Fu. Power-up sram state as an identifying fingerprint and source of true random numbers. *IEEE Transactions on Computers*, 58(9):1198–1210, 2009.

[16] Chi-En Yin and Gang Qu. Improving puf security with regression-based distiller. In *Proceedings of the 50th Annual Design Automation Conference*, page 184. ACM, 2013.

[17] Chi-En Yin and Gang Qu. Temperature-aware cooperative ring oscillator puf. In *Hardware-Oriented Security and Trust, 2009. HOST'09. IEEE International Workshop on*, pages 36–42. IEEE, 2009.

[18] Zdenek Paral and Srinivas Devadas. Reliable and efficient puf-based key generation using pattern matching. In *Hardware-Oriented Security and Trust (HOST), 2011 IEEE International Symposium on*, pages 128–133. IEEE, 2011.

A Silicon PUF Based Entropy Pump

Qian Wang and Gang Qu

Abstract—The security level of many cryptographic protocols and secure systems is determined by the strength of the cryptographic keys, which can be measured by entropy. Finding an entropy source that can generate secure keys with high entropy is a very challenging problem and it is normally associated with high cost. Instead of looking for a low-cost entropy source, we study in this article how to improve the entropy generated by a low-entropy source. Unlike the existing approaches based on hash function or cryptographic protocols, our solution leverages the intrinsic randomness in physical properties such as silicon physical unclonable functions (PUFs). Silicon PUF is a piece of circuitry that can capture certain intrinsic on-chip variations that were introduced during the chip fabrication process. It is generally believed that such variations are random and unpredictable. In this article, we demonstrate that the silicon PUF can be used as an effective entropy pump to boost low-entropy keys. Our approach is based on a recently developed highly flexible ring oscillator (RO) PUF. When we use the low-entropy key to configure the RO PUF, we find that the corresponding PUF response exhibits higher entropy, which means that the key's entropy has been improved. We implement our design on Nexys 4 Artix-7 FPGA board and demonstrate that the configurable PUF structure can successfully enhance the entropy of input keys. Compared to the other entropy enhancement methods, our PUF based entropy pump has the lowest hardware cost. Moreover, we apply this in a password enhancement application to provide robust high entropy passwords that can resist attacks such as the pre-compute attack.

Index Terms—Silicon PUF, entropy, cryptographic keys, password enhancement

1 Introduction

The importance of entropy to information security can be seen as it is a standard metric for the randomness of cryptographic keys [1]. Pseudorandom number generators (PRNGs) are commonly used to generate keys, but the sequence of numbers they generate is deterministic and can be traced predictably to the seed. Thus PRNGs must be seeded with sufficient entropy from a reliable source [2]. Entropy sources that provide true randomness are usually based on non-deterministic physical processes, such as ring oscillators, unpredictable events, human-driven mouse movements or keyboard stroke timings. However, these sources often provide a limited amount of unpredictability or low entropy. The development of quantum information science offers high entropy sources built on photonic and atomic qubits where the randomness depends on quantum effects in physics. But the cost of implementing silicon quantum wire arrays to test the quantum phenomenon is extremely expensive compared to the traditional way to collect entropy [3]. The goal of this article is to develop a low-cost platform to enhance the entropy of keys generated from low entropy sources so they can be used for security applications. Unlike the existing approaches that rely on encryption, hash function, or information-theoretic techniques [4], [5], we propose to leverage the intrinsic randomness of physical features in the system and we demonstrate this through the example of silicon physical unclonable function (PUF).

PUF is a kind of innovative circuits that can extract the fabrication variation from hardware characteristics of integrated circuits (ICs) during its manufacture process [6], [7]. Due to its properties of low-power, small area, unpredictability, and unclonability, PUF-enabled secure architectures have been proposed for the authentication of individual ICs and the generation of volatile secret keys for cryptographic operations, which are best suited for the Internet of Things (IoT) applications where resources are extremely constrained. In this article, we study how PUF can be used as an entropy pump to boost the entropy of cryptographic keys.

The output of an entropy source often passes through a pseudorandom function (PRF) conditioner, such as a hash function or a block cipher, to distribute entropy uniformly across the bits of outputs samples. However, the hash function has some limitations when using to distribute entropy, as the main drawback of those algorithms is that the attacker could pre-compute the hash values for the input seed [8]. Such attacks always exist in the password system, because the password system only stores the hash value of the password. Therefore, an attacker can pre-compute the hash values for common passwords variants. This enables a very quick recovery of passwords whose length is short. As an example of our proposed system, we demonstrate how our silicon PUF based entropy pump can resist such rainbow attacks. The proposed PUF based entropy pump uses the ring oscillator PUF, instead of the hash function, to build a password system. First of all, the embedded randomness of the PUF could serve as a security enhancer which could pump the entropy source, thus offering security applications with higher entropy. For example, in the key generation procedure, higher entropy source could provide corresponding higher security level of

• *The authors are with the Department of Electrical and Computer Engineering, University of Maryland, College Park, MD 20770. E-mail: {qwang126, gangqu}@umd.edu.*

Manuscript received 31 July 2017; revised 6 Oct. 2018; accepted 11 Oct. 2018. Date of publication 19 Nov. 2018; date of current version 10 May 2019. (Corresponding author: Qian Wang.)
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TDSC.2018.2881695

keys thus lower the risks for entropy loss in the repetition code configurations. Second, the physical properties also assemble the PUF with uniqueness between different devices which could be utilized in the authentication. For example, if the attacker pre-computes the passwords on one device, the pre-computed table will become useless for other devices, which significantly hinders the attacker's ability to implement attacks on a large number of devices of the same type.

We propose a novel configurable PUF structure to achieve the goal to enhance the entropy of the input source. To make our proposed PUF robust, we choose to implement flexible configuration delay based RO PUF and develop the robust inverter selection algorithms to assist our hardware structure. We develop three mapping strategies to select an odd number inverters out of the chain to oscillate. Our structure works for different input cases (from low entropy to high entropy). And we test the entropy results both in simulation environment and on FPGA board. To demonstrate our PUF based structure works for real-world application, we develop the password strength machine and prototype it in hardware.

In summary, this paper presents the following contributions:

(1) We design and implement the entropy enhancer based on the RO PUF. As a part of our implementation, we also show how to use intrinsic variation of PUF to enhance the entropy of source and achieve security applications.
(2) We study the trade-off between randomness and stability of the configurable RO PUF under different temperatures. Based on this, we propose a modified structure which provides stability at the cost of losing some amount of entropy (around 0.08 on average). We show an application uses the response of our PUF scheme to enhance the password's strength. This scheme would play an important role in security applications such as IoT authentications where the devices are deployed in temperature changing environments.

2 Preliminaries

2.1 RO PUF and Configurable RO PUF

PUF is a lightweight hardware primitive which exploits the inherent manufacturing variations to generate unclonable secrets. The RO PUF is one of the popular PUF designs that leverages the delay difference of two ring oscillators [7]. An RO is a device composed of an odd number of inverters, whose output oscillates between two voltage levels in a fixed frequency. One secret bit can be generated from a pair of ROs by comparing their frequencies and the bit is considered as random and not predictable because the delay distribution is random due to the fabrication variation. The single component generating the delay is usually called delay element or stage which has inverters or other logic gates. The name as k-stage PUF refers to the structure who has k delay elements for a single RO ring.

The notion of reconfigurability in RO PUF has been introduced by Maiti et al. [9]. In their approach, at each stage of the RO, they place two inverters and use a multiplexer (MUX) to select one to form the RO. Gao et al. [10] improve this design by using only one inverter in each stage and replacing the other inverter by wire. The input configurable bit actually controls the multiplexer thus the selection. If the selection bit is '1', the corresponding inverter in the stage would be used in the ring; if the selection bit is '0', the inverter will be disconnected and the signal will pass through the wire. In this article, we explore a new application of the configurable RO PUF in boosting the entropy of the input bit-stream.

2.2 Definition of Entropy

Entropy in information theory is defined as the function of probability distribution which is a measurement of the unpredictability of information content [11]. The entropy of a random variable X with probability mass function $\Pr(x)$ is defined by:

$$H(X) = -\sum_{x} \Pr(x)\log_2 \Pr(x). \tag{1}$$

For a random variable X with binary outcomes (0 or 1), if it is distributed uniformly, the entropy $H(X)$ should be 1. The conditional entropy is:

$$H(X|Y) = -\sum_{y\in Y} \Pr(Y=y)H(X|y). \tag{2}$$

The joint entropy is:

$$H(X,Y) = -\sum_{x\in X}\sum_{y\in Y} \Pr(x,y)\log \Pr(x,y). \tag{3}$$

The chain rule of entropy based on conditional entropy and joint entropy is:

$$H(X,Y) = H(X) + H(Y|X). \tag{4}$$

More generally the entropy of a collection of random variables is the sum of the conditional entropy:

$$H(X_1, X_2, ..., X_n) = \sum_{i=1}^{n} H(X_i|X_{i-1}, ..., X_1). \tag{5}$$

By the chain rule of entropy, we have the following inequality property for joint entropy:

$$H(X_1, X_2, ..., X_n) \leq \sum_{i=1}^{n} H(X_i). \tag{6}$$

This becomes an equality if and only if X_i are independent. We will utilize this equality to calculate the joint entropy.

The entropy definitions discussed above are all Shannon entropy. In the analysis of cryptographic constructions, min-entropy is also widely used as the most conservative way to measure the unpredictability of a set of outcomes. Formally, the min-entropy for a random variable X is:

$$H_{min}(X) = -\log_2(\max_{x\in X} \Pr(X=x)). \tag{7}$$

In this paper, without specific mention, we use Shannon Entropy.

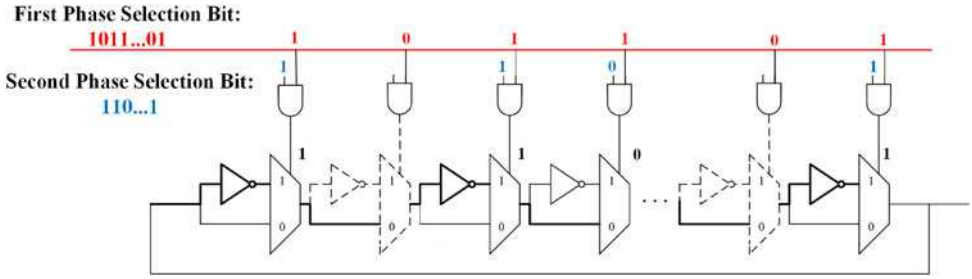

Fig. 1. Architecture of the two-phase configurable RO (The bold line shows the selection path in this configuration bit.).

2.3 Entropy of PUF

The concept of PUF entropy was introduced to measure the unpredictability of PUF readouts [12]. In [13], the author introduced several methods to model PUF and provided an approach for entropy estimations. Paper [14] derived the tight upper bounds on the min-entropy of several PUFs including the RO PUF. We will refer their conclusion to support the entropy enhancement for our approach. More specifically, a relationship between the stage m of PUF and min-entropy provided by the PUF is proved in the paper. As the formula shows when m is odd,

$$H(X) \leq -\log_2\left(\frac{1}{2}\left(1-\sqrt{\frac{m-1}{m}}\sum_{i=0}^{(m-3)/2}\frac{(2i)!}{(i)^2(4m)^i}\right)\right). \quad (8)$$

Based on this inequality, we can calculate that a 63-bit input PUF could generate around 200 bits output entropy and the 127-bit input would generate around 500 bits of entropy. This implies that output entropy from PUF has the potential to be much higher than the entropy of the challenge input, which motivates our work on using RO PUF to enhance entropy.

3 Design of Novel RO PUF Structure

Our proposed configurable RO structure is derived from the basic RO PUF concept by adding the multiplexer to make the inverters selectable, as shown in Fig. 1. Notice that we only draw one ring in the figure for simplicity, although the RO PUF has two parallel rings. The motivation of our design is to build an entropy booster on the proposed oscillated rings. Thus, our design has some significant differences from the basic RO PUF. For instance, we come up with a new selection method for the configurable input to make the output random, as well as to improve the output entropy eventually. Unlike the configurable RO PUF design in [10] which is to make the response stable, our design aims to extract the uncertainty in the PUF output and utilize this on the security primitive. Such a design needs the delay difference of inverters to be small to have a high uncertainty. To achieve this purpose, we propose two phases of configuration for the RO, which is illustrated in Fig. 1 and explained in the following.

3.1 First Phase Design: Inverter Selection Method

The first phase design strategy of the PUF is to select inverters from the ring which have small delay variation. In [10], the authors choose those inverters which have the largest delay difference to guarantee the stability of the output when the environmental conditions change. However, our strategy goes in the opposite direction as to select inverters potentially in providing randomness of the PUF output. This requirement drives us to select those inverters making the delay differences of the two rings small. By doing this, the output would have a lower dependence on the intrinsic sum of delays. We do not expect the output bit to be fixed on whether '1' or '0'. As a result, if the delay difference of two rings is small, choosing different configuration challenges makes the output random. On the contrary, if the delay difference of two rings is large, the final comparison result might be fixed to 0 or 1, no matter what is the challenge input. This situation implies less of randomness and will hinder the application for entropy enhancement.

Algorithm 1. Inverter Selecting Algorithm

Input: n-element inverter arrays: $Inv_1(1:n)$, $Inv_2(1:n)$
Output: k-element inverter arrays: $Inv_1(1:k)$, $Inv_2(1:k)$
1: Sorted Inv_1 , Inv_2 in descending order
2: **for** $i = 1$ to $n-k$ **do**
3: **if** sum(Inv_1) $\leq$ sum(Inv_2) **then**
4: remove $Inv_1(1)$ and $Inv_2(end)$;
5: **else**
6: remove $Inv_1(end)$ and $Inv_2(1)$;
7: **end if**
8: rearrange $Inv_1(1:n-i)$ and $Inv_2(1:n-i)$
9: **end for**
10: **return** $Inv_1(1:k), Inv_2(1:k)$

To implement this, we first measure the delay of each stage, which consists of the delay of the inverter and the multiplexer, following the method proposed in the paper [10]. Next, we will select a set of suitable inverters according to the measurement from the former step, the details for the selecting algorithm is described in Algorithm 1. The preliminary input of the algorithm is the delay of each stage measured and stored in the arrays represent as Inv_1 or Inv_2. First, we sort the inverters by the delay in descending order and define k $(k < n)$ as the number of inverters to be selected. Then, we remove the inverter out of the ring one by one following the routine in Algorithm 1 until we have k inverters left. For example, if $sum(Inv_1) > sum(Inv_2)$, we will remove the inverter that has the largest delay in Inv_1 and the smallest delay in Inv_2. 'Remove' in real implementation refers physically isolated, which means that inverter would not be used in the future. Since the delay of each inverter is similar, the delay difference between two rings decreases after this rearrangement. After repeating this process $(n-k)$ times, the remaining inverters in the two rings will have a smaller delay difference which fulfills our goal to make the PUF output random. The result after the first

phase selection is also illustrated in Fig. 1. As the figure shows, the inverter drawn in dashed lines are those removed by the first phase. Those inverters would not be used in the next phase selection.

3.2 Second Phase Design: Odd Number Selection

An odd number of inverters should be guaranteed to trigger the oscillation in the RO ring. However, as the configuration input is generated from the random source, we cannot guarantee that the input bit-stream always consists of odd number of '1' . This implies that we should implement an approach forehead to map the configuration input to a bit-stream whose Hamming weight is odd. One native solution is designing a strategy to map all the input combinations to the values which meet the requirement. The obvious drawback of this method is that the mapping strategy cannot guarantee uniformity of the outputs. That is to say, it is difficult to make the probability of each inverter to be chosen equally. To overcome the biased mapping problem, we leverage a simple, yet efficient back-up inverter technique by adding a back-up inverter at the end of the ring to adjust the Hamming weight based on the parity of the input. This method also balances the chosen probability of each inverter. We will describe both the mapping and back-up inverter strategies in details in the following.

3.2.1 Mapping Strategy on Configuration Bits

The first straightforward method to solve the odd-number problem is to map any inputs to the values with the odd number of '1' s. We design a simple look-up table to achieve this. Because the probability of each bit cannot be identical in the mapping strategy, the mapping will inevitably introduce bias to the randomness. Besides, if the mapping strategy is too complex, the time delay and power consumption will affect the efficiency of the system. To reduce the cost and complexity, we choose small odd numbers (3, 5 or 7) and design two strategies as 3 to 5 mapping and 7 to 5 mapping.

3 to 5 mapping: we map every 3-bit input to a 5-bit vector, which must contain three '1' s. There are in total $2^3 = 8$ input cases for a 3-bit binary vector. Choosing 3 positions in the 5 bits, we will get $\binom{3}{5} = 10$ output cases. Therefore, this mapping strategy is to map 8 input cases to 10 output cases. Obviously, there are two unused output cases which cause the probability for each bit is not equal.

5 to 7 mapping: we plan to map a 5-bit input to a 7-bit output vector, which has the Hamming weight of 5. The total number of input cases is $2^5 = 32$ and there are in total 21 output cases of when we choose 5 positions in the 7-bit vector. Unlike the 3 to 5 mapping strategy discussed above, the 5 to 7 strategy has input redundancy. There are 11 pairs of two 5-bit vectors that will map to the same 7-bit configuration vector. We deliberately design this mapping table, balancing the selecting probability of each inverter. As a result, we map those two vectors that are complementary in binary to the same output vector. For example, both 5 (00101) and 26 (11010) will map to the same vector (1110101).

3.2.2 Back-up Inverter Strategy on Configuration Bits

The drawback of those mapping strategies is that they cannot avoid the redundancy in the mapping, which will influence the randomness. For example, the 3 to 5 mapping has output redundancy and the 5 to 7 mapping has input redundancy. To solve the mapping redundancy problem, We come up with another effective method called the back-up inverter strategy to meet the requirement for odd number of inverters. For example, if we have an 8-bit input, we will add another inverter (9th) in the ring as the back-up inverter marked as INV*. If the Hamming weight of the 8-bit input is odd, which could generate the oscillation, we will make the corresponding selection bit of INV* '0'. Otherwise, if the input vector has selected even number of inverters that could not lead oscillating, we will add the last inverter as INV* in the ring to generate the output bit. First, this strategy solved the unbalanced mapping problem because the selection bit of the INV* is decided by the parity of the input stream which is uniformly distributed in nature. In addition, this strategy has a great advantage in hardware implementation, as the last configuration bit for INV* could be easily got by bit exclusive-or the input.

3.3 Randomness versus Stability

As we discussed above, our proposed entropy pump can be designed for different applications based on both the randomness and stability feature in it. That is to say, we can manipulate the property of the PUF by designing different inverter selection algorithms (The algorithm for the first phase). To our best knowledge, the former selection algorithms in literature are always aiming to guarantee the stability of PUF output [10]. Here, we develop a corresponding algorithm to achieve the random output instead, thus generating higher entropy, shown in Algorithm 1. However, due to the delay of each inverter would be drifted by the temperature or voltage supply, the stability comes to be another important concern when we would like to apply our structure in the authentication applications, e.g., the password application. In this kind of security applications, both the stability and entropy should be considered. As a result, we come up with a modified algorithm, shown as Algorithm 2, which guarantee the stability of the PUF under different temperatures with an acceptable entropy loss.

Algorithm 2. Stable Inverter Selecting Algorithm

Input: n-element inverter arrays: $Inv_1(1:n)$, $Inv_2(1:n)$
Output: k-element inverter arrays: $Inv_1(1:k)$, $Inv_2(1:k)$
1: Calculate difference $diff(1:n) = Inv_1 - Inv_2$
2: Group the $diff$ in positive and negative as $diff_p$ and $diff_n$, and sorted in decreasing order by absolute value as $diff_p(1:n_1)$, $diff_n(1:n_2)$, $n_1 + n_2 = n$
3: **for** $i = 1$ to $n - k$ **do**
4: **if** sum(Inv_1) $\leq$ sum(Inv_2) **then**
5: remove the inverter pairs at the index n_1 as $diff_p(n_1)$, $n_1 = n_1 - 1$
6: **else**
7: remove the inverter pairs as the index n_2 for $diff_n(n_2)$, $n_2 = n_2 - 1$
8: **end if**
9: rearrange $Inv_1(1:n-i)$ and $Inv_2(1:n-i)$
10: **end for**
11: **return** $Inv_1(1:k), Inv_2(1:k)$

Algorithm 2 is designed to select those qualified inverters in the ring under two conditions: first, it would make the delay difference of the two rings smaller, the same as Algorithm 1; second, it will make the difference of two inverters in the same position large. After this selection algorithm, the left inverters would hold the small delay differences between two rings which makes the output random. Meanwhile, compared between two rings at the same stage, the delay difference is large, which guarantees the response to be stable.

4 Simulation Results and Analysis

We first test our new PUF structure as an entropy pump on a simulated dataset to verify the assumption for entropy enhancement.

4.1 Input Vector Generation

We generate several input test vectors (the probability of '1' occurs from 0.1 to 0.9) to test the proposed entropy pump. More specifically, the input bit-streams are generated following Bernoulli distribution $B \sim (n, p)$. For example, the first test case is acquired when the probability is set to $p = 0.1$. The input entropy of this case is low because of lacking randomness. With probability $p = 0.5$ as both '1' and '0' occur at 50 percent. The input entropy is high in this case, which reaches the full binary entropy as 1.

4.2 Simulation Entropy Results

Our proposed entropy pump is first demonstrated on the simulated RO PUF with delay measurement obtained from the Virginia Tech's public PUF dataset, which consists of frequency measurements of ROs from 198 Xilinx Spartan (XC3S500E) FPGA boards. We assume that the delay of RO measured in the dataset is proportional to the delay of inverter. Thus, we treat the delay of RO as the delay of the inverter in our simulation. Among the 198 boards, 194 boards have the measurements at a fixed supply voltage (1.20V) and a fixed temperature ($25°C$). We simulate our proposed PUF structure using data from these 194 boards and extract the output of PUF for the following entropy test.

We implement the reconfigurable PUF for the three mapping cases discussed above:

Case 1: 3 to 5 mapping strategy has been used.

Case 2: 5 to 7 mapping strategy has been used.

Case 3: back-up inverter strategy has been used.

Here, we use Shannon entropy to evaluate both the input and output. For simplicity, we keep the length of output consistent to the length of configuration input. Thus, we place the same number of RO pairs as the number stages of one RO. The input is defined as $X = (X_1 X_2 ... X_k)$ and the output of the RO PUF is defined as $Y = (Y_1 Y_2 ... Y_k)$. For each time, we select one challenge vector as x, and we will get an outcome vector in the same length as y. First, we calculate the Shannon Entropy of each bit X_i as $H(X_i)$. Then we utilize the equality property of the chain rule to get the joint entropy of random variable X as $H_{in}(X) = \sum_{i=1}^{k} H(X_i)$ based on Eq.6, as well as for the output variable Y as $H_{out}(Y) = \sum_{i=1}^{k} H(Y_i)$. Note that the equality holds when the X_i (or Y_i) are independent.

The Shannon entropy results for the input and outputs with three mapping strategies are all illustrated in Fig. 2.

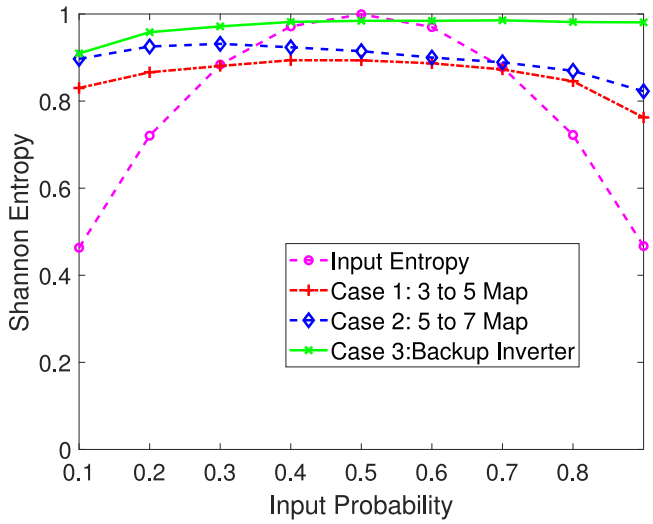

Fig. 2. Entropy results for different mapping strategies.

We can observe that in all the three cases, the output entropy has been significantly improved. As the figure shows, when the input is not random and the entropy is below 0.5, the output entropy has been pumped up to 0.9 (near the full entropy). However, when the input entropy is already high which is close to 1, the output entropy of mapping strategies (Case 1 and Case 2) decrease 10 percent from the input entropy, that is reasonable because the mapping strategies have bias mapping as we discussed before. However, the back-up inverter strategy (Case 3) could partially solve this problem as its entropy is close to the full entropy.

4.3 XOR with the Original Input

From the simulation results, we observe a large enhancement on the output entropy when the input entropy is low. However, when the input has high entropy, our scheme may reduce the entropy. To overcome this, we propose to exclusive-or the output with the initial input. This solves the problem when the input entropy is already near the full entropy, the XOR output could keep the high entropy as well. This scheme is the same as the one-time pad in cryptography. In the following, we will prove the bound for improvement in entropy based on the one-time pad property and the definition of perfect secrecy system. It is believed that the one-time pad scheme is perfect secrecy [15]. We define the input as a random variable X, the output Y and the result of the exclusive-or is defined as Z. Therefore, the final output entropy would be $H(Z)$, where $Z = X \oplus Y$. The definition of conditional entropy could be written as $H(X|Y)$. It measures the average uncertainty about X given the observation of the variable Y. For a secrecy system, we call the conditional entropy $H(K|C)$ as the key equivocation. And it holds the theorem below [15].

Theorem 1. $H(K|C) = H(M) + H(K) - H(C)$

As an immediate implication of the above theorem, we have $H(K|C) = H(K)$ in the case of a perfect secrecy system, and this is a necessary and sufficient condition for perfect secrecy. That is, uncertainty about the key does not decrease with knowledge of the ciphertext. We plug in our

TABLE 1
Results of Input Entropy versus Output Entropy

Input probability	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Input Entropy H_{in}	0.463	0.721	0.884	0.972	0.999	0.97	0.878	0.722	0.467
					Simulation Results				
Output Entropy H_{out}	0.9095	0.9583	0.9716	0.9818	0.9844	0.9844	0.9855	0.9816	0.9808
Output Entropy H_{xor}	0.9398	0.9864	0.9949	0.9984	0.9992	0.9991	0.9980	0.9928	0.9880
					Implementation Results				
Output Entropy H_{out}	0.8879	0.9327	0.9465	0.9543	0.9608	0.9584	0.9527	0.9385	0.8822
Output Entropy H_{xor}	0.9289	0.9722	0.9905	0.9963	0.9972	0.9943	0.9862	0.9678	0.9151

definition to the Theorem 1 as $M = X, K = Y, C = Z$, from this, we could have

$$H(Y|Z) = H(X) + H(Y) - H(Z). \tag{9}$$

From the conditional entropy definition, we have $H(Y|Z) \leq H(Y)$. Hence

$$\begin{aligned} H(X) - H(Z) =& H(Y|Z) - H(Y) \leq 0 \\ H(X) \leq& H(Z). \end{aligned} \tag{10}$$

Thus, it proves that we can improve the exclusive-or entropy $H(Z)$ from the input entropy $H(X)$.

The results for the input entropy, output entropy and the exclusive-or entropy for 9 different input cases are shown in the middle part of Table 1. We can observe that the original output entropy (H_{out}) has been improved to close to the full entropy 1. Also, the exclusive-or entropy (H_{xor}) increased by 0.02 than the H_{out}.

4.4 Improve the Entropy by Iteration

By using the XOR method discussed above, we could achieve almost full entropy when the input entropy is high (i.e., when the probability is from $p = 0.3$ to $p = 0.7$), demonstrated in Table 1. However, we find out that there is still a small gap to the full entropy (around 0.05) for the other 4 input cases (i.e., $p = 0.1, p = 0.2, p = 0.8, p = 0.9$), whereas the input entropy is too low and only one-time enhancement cannot pull it up to the full entropy. Intuitively, iterating the result to the PUF pump would further enhance the entropy. We conduct the iteration with the four low-entropy input cases list above. Results in Fig. 3 show that by iterating at least 4 times, the output reaches to full entropy.

5 FPGA Implementation and Results Evaluation

5.1 Implementation

We implement our novel RO PUF design for entropy enhancement on the Xilinx Artix-7 FPGA in Nexys 4 DDR board. The FPGA board communicates with the PC by the serial port. Through the serial port communication, We send the configuration input generated by the Matlab to RO array implemented on FPGA board and collect the corresponding response. We choose to implement the back-up inverter strategy on the FPGA board as it has the best simulation result shown in the former section. Another reason is that the parity bit for the last inverter is easier to implement in hardware by using XOR gates. We implement 32 pairs of RO in total and each ring consists of 32+1 stages (32-bit configuration input and 1-bit for back-up inverter). The layout of the RO obeys the relative location constraints. Also, this constrains could help us to reduce the delay difference between the two parallel rings, thus making the output random. The inverters and multiplexers are implemented using the basic LookUp Table (LUT) element in FPGA. And in total 608 LUTs (569 slices) are used in the RO design for 32 groups and each group has 32+1 stages.

5.2 Entropy Result from FPGA Implementation

The input and output entropy results acquired from the FPGA board are shown in the bottom in Table 1. The results of FPGA implementation are consistent with the simulation results we got before but a bit lower than the simulation. The average output entropy for 9 cases is 0.9349 in FPGA implementation and the corresponding entropy in simulation under the same condition is 0.9717. Meanwhile, H_{xor} from FPGA is 0.9721 and it is 0.9885 in simulation. As the table shows, when the input entropy is lower than 0.9, our design could improve the entropy by around 0.03. However, the same entropy decrease exists when the input entropy is already high. But by XOR with the input source, the final entropy goes back to near full entropy 1.

5.3 Dependence and Bias

We test the dependence and bias of our structure for 5 input cases with probability as 0.1, 0.3, 0.5, 0.7 and 0.9 respectively.

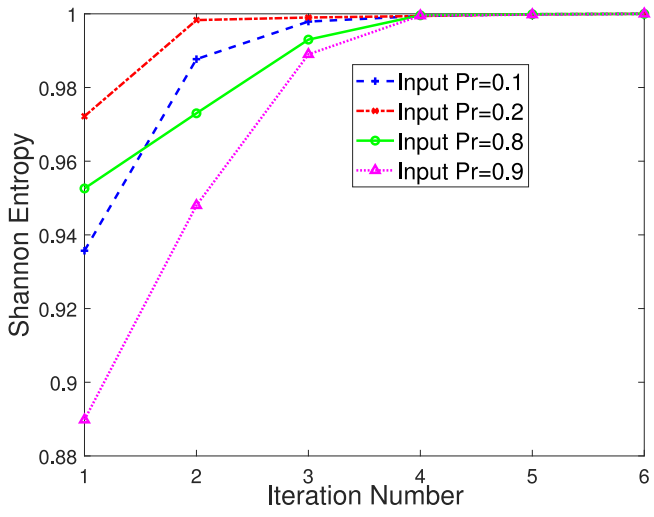

Fig. 3. Entropy results of iteration with different probability.

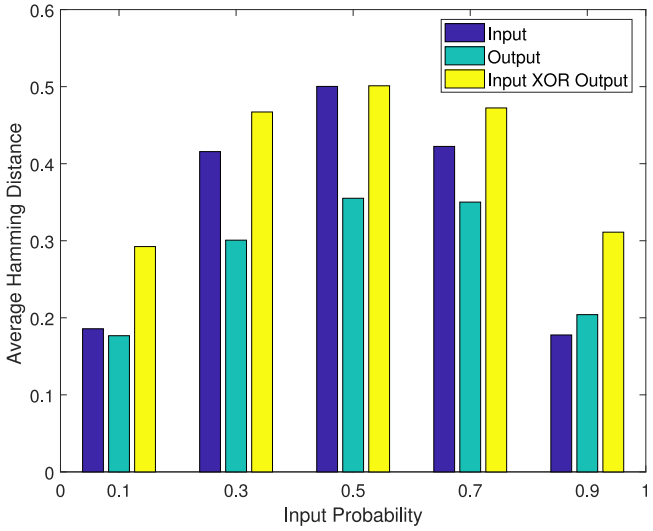

Fig. 4. Dependence distribution for different input cases.

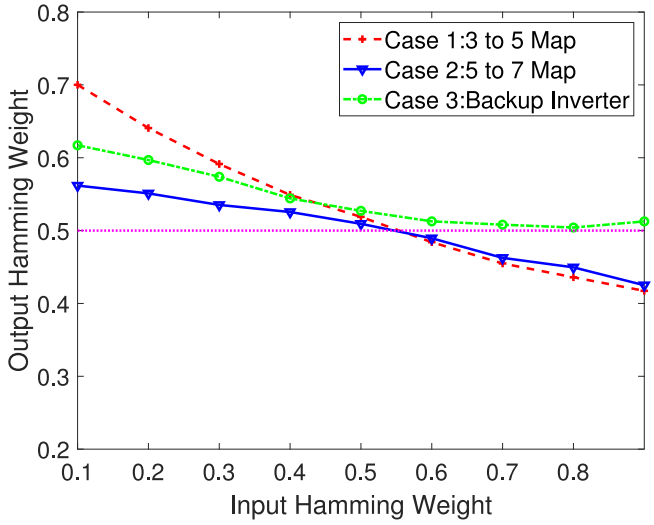

Fig. 5. Hamming Weight for different strategies.

5.3.1 Configuration Dependence

We define the configuration dependence (CD) as the average Hamming distance of the two responses from any two different challenges. For k random selected challenges, L-bit PUF responses are generated, i.e., $r_1, r_2, ..., r_k$. The configuration dependence is calculated as follows:

$$CD = \frac{2}{k(k-1)} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \frac{HD(r_i, r_j)}{L}, \tag{11}$$

where $HD(r_i, r_j)$ denotes the Hamming distance between two responses as r_i, r_j.

$$HD(r_i, r_j) = \sum_{m=1}^{L} (r_{i,m} \oplus r_{j,m}), \tag{12}$$

where $r_{i,m}$ and $r_{j,m}$ are mth bit of L-bit r_i and r_j respectively. This metric indicates that whether the responses generated by different challenges are related or not. We send $k = 1000$ random selected challenges to the PUF structure and calculate the Hamming distance for any two outputs. Then the corresponding CD is got by averaging the Hamming distances. Intuitively, the average Hamming distance of two random vectors should be half of the vector length, $L/2$ in this case. Thus, the ideal CD by averaging all the responses should be 0.5 based on Eq.11. As shown in Fig. 4, the configuration dependence of the output bit-stream is around 0.3 which is lower than the ideal value as 0.5. This is because the correlation of stages in the PUF influences the randomness of output. To overcome this drawback, we exclusive-or the input with the output bit-stream and the XOR output result goes back to ideal value as shown in Fig. 4.

Results show that for all the 5 test cases the dependence of XOR output is better than the original output. In the most bias input case, the input dependence is as low as 0.2. However, the original output CD does not improve too much. The CD of the XOR output increases a lot in this case.

5.3.2 Hamming Weight Calculation

Now we focus on the Hamming weight as the bit-wise correlation of the bit-stream. If the probability of '0' and '1' occurs half to half in the vector, the average Hamming weight per bit, in this case, should be around 0.5 which shows no bias for the inter-bit. Generally, the formulation to calculate the Hamming weight is shown as follows,

$$HW = \frac{1}{k} \sum_{i=1}^{k} \frac{1}{L} \sum_{m=1}^{L} r_{i,m}. \tag{13}$$

The same notations are used as before: k is the number of responses and L is the bit-length of each response. $r_{i,m}$ represents the m-th bit of L-bit response r_i. The bias of input bit-stream changes with the probability because each bit of input is generated as an independent random variable, so the average Hamming weights of the input stream is linear to the probability as shown in the as x-axis of Fig. 5. The y-axis in the figure illustrates the Hamming weights of output bit-streams for three different mapping strategies we discussed above. The dashed line shows the ideal value 0.5 which means for all the bits in the response, '0' and '1' occurred evenly. The figure shows that the output Hamming weight swings between the ideal value. This trend is related to the correlations of intrinsic delay of the oscillated rings. Due to the layout distributions of rings, when the input configuration is not random, the rings may generate the same response highly relates to the intrinsic delay but not too much depends on the configuration input. As a result, the Hamming weight of the corner cases ($p = 0.1$ or $p = 0.9$) are away from the ideal value. At last, from Fig. 5, we could find out that the back-up inverter strategy has the best result as it is the closest to the ideal value.

5.4 Results of Comparisons between Randomness and Stability

As expected, different temperature conditions yield different entropy output. It can be seen from Fig. 6 that the average entropy for the Algorithm 1 (random algorithm) is around 10 percent higher than the entropy of Algorithm 2 (stable algorithm), which is consistent with the algorithm. It is proved that under the same input

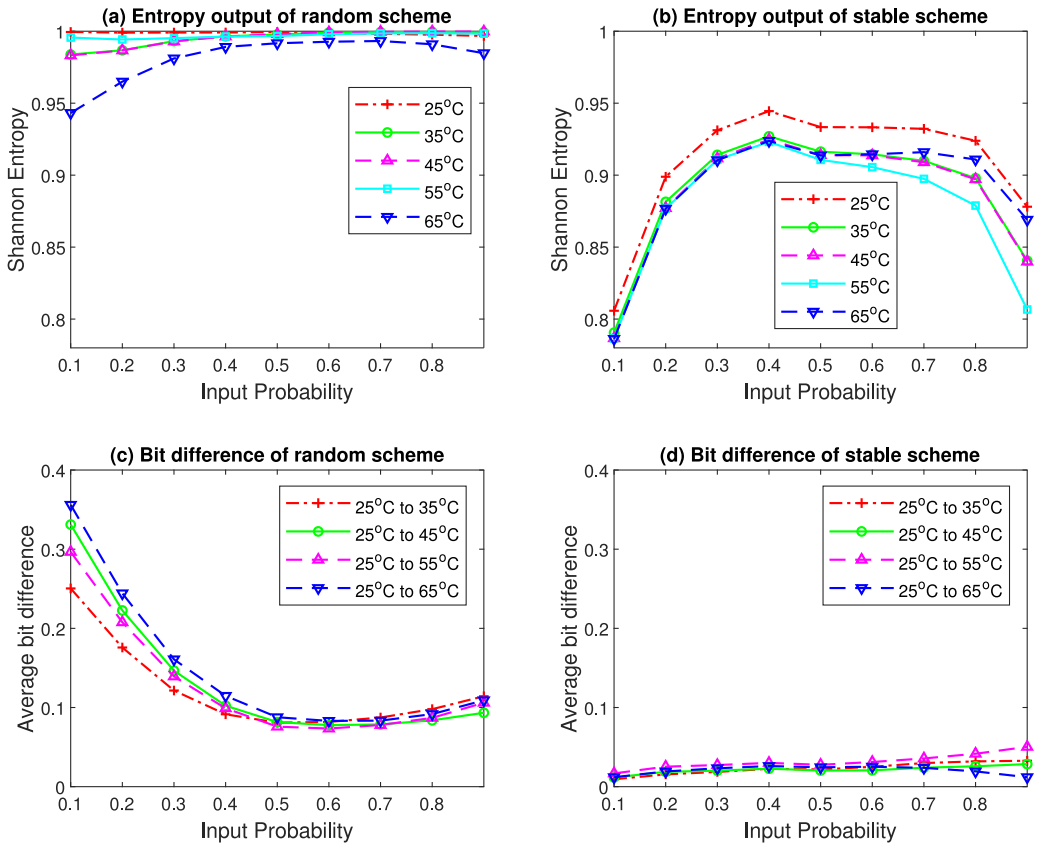

Fig. 6. Entropy and bit difference results of random and stable schemes.

condition, the entropy result from Algorithm 2 drops a bit to get more stability during temperature changes. Besides, the temperature also drifts the entropy because our selection algorithm is applied at the fixed temperature as 25°C. When the temperature is higher, the delay of inverter might change from the original delay. It will influence the frequency of rings and eventually change the response bit. For this test, we choose the response bit at 25°C as the reference to study the influence of different temperatures. Compared the lines in Fig. 6c and 6d, we could see that the average bit difference is different for two schemes, as for Algorithm 1 is 15 percent and 2 is 3 percent. It demonstrates our expectation that the Algorithm 2 is much stable than the Algorithm 1 when the environment changes. Meanwhile, a variation of the bit difference for different temperatures across 25°C to 65°C are also plotted in Fig. 6. It shows the variance of bit difference is also smaller for Algorithm 2 than Algorithm 1.

5.5 NIST Statistical Test Results

Besides, we evaluate the randomness of bit-streams using the NIST statistical test suite. NIST test suite is a statistical package consisting of 16 tests that were developed to test the randomness of binary strings. Table 2 shows our experimental results in the NIST suite as well as the overall passing proportion. The pass proportion illustrates how many tests the stream passed over the total 16 sub-tests. The P-value represents the probability of obtaining a test statistic as large or larger than the one observed if the sequence is random. Hence, small values (conventionally, P-values < 0.05 or P-values < 0.01) are interpreted as evidence that a sequence is unlikely to be random. The P-value in the last line of the table is the sum of all the 16 tests.

5.6 Security Analysis: Modeling Attack

Modeling attack is reported the most efficient attacks on PUFs, especially for delay-based PUF. The attacker makes

TABLE 2
Results of NIST Test Suite

Input probability		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Input Test Result	Pass proportion	0/16	1/16	3/16	1/16	16/16	5/16	3/16	2/16	1/16
	P-Value	0	0.72	1.03	0.75	7.36	2.63	2.41	1.92	1
Output Test Result	Pass proportion	2/16	2/16	4/16	3/16	7/16	6/16	4/16	3/16	2/16
	P-Value	1.46	0.31	2.23	1.91	2.65	2.25	0.78	0.88	0.22
XOR Output Test Result	Pass proportion	3/16	3/16	9/16	10/16	16/16	8/16	6/16	3/16	1/16
	P-Value	0.99	2.52	2.32	6.24	6.96	4.66	2.31	1.29	0.45

use of the publicly accessible interface to collect a large number of challenge-response pairs (CRPs) in order to model the PUF. Modeling attacks have been demonstrated successfully in modeling delay based PUFs (e.g., the arbiter PUF and RO PUF) given around 1000 CRPs. Our proposed PUF structure is susceptible to modeling attacks if the CRPs can be observed. However, in our article, the PUF is proposed to use as an entropy pump not used in the traditional authentication. As a result, there is no direct access interface to read the input and output from the PUF in this application. Besides, to get enough number of valid CRPs is challenging for the attacker as he needs to break in the chip and get the input and output in a pair. Literature has demonstrated that if CRP access interface is not guaranteed, the modeling attacks cannot be successfully launched [16]. In addition, the selection algorithms are reconfigurable in our proposed structure. As we demonstrated before, different selection algorithms have been applied in the PUF (i.e., Algorithms 1 and 2). If the configurable algorithm changes, the former obtained CRPs cannot be used in modeling which reduces the risk of attacks [17].

6 Application for Password Strength Enhancement

We have already demonstrated that our proposed PUF structure could improve the entropy of the given input sources. Based on this property, we apply the entropy pump for password strength enhancement in IoT related field.

On today's Internet, websites always authenticate users by requiring a password and the IoT scale authentication also uses the concept of passwords. Because of the limits of passwords' length, Id/password pairs are relatively lightweight, yet managing them in high quality is more challenging. Unfortunately, bad passwords cause even worse effects to IoT-scale authentication than the Internet-scale authentication because of the restricted storage and bandwidth of IoT devices. Another concern is that the credential of the password is easy to be thwarted by malicious attacks. For example, the quality of the passwords has been proved to be lack of entropy and the passwords are easy to be hacked by attackers. Classical cryptography algorithms have provided sound solutions in solving these problems related to passwords. However, for most proposals are designed for Internet-scale applications which either need post-quantum cryptography or expensive cryptography algorithms (i.e., Diffie-Hellman protocols) [18], [19]. Moreover, in paper [20], they also propose to use a hardware security module (HSM) at the authentication server to prevent off-site password discovery. While their scheme is designed for the Internet scale as they try to protect the stored user passwords in the server from cracking by the adversary. Therefore, to some extent, those schemes cannot meet both the security and resource requirements from the compact IoT devices which have strict energy and timing restricts. Hence, our proposed PUF based entropy pump can serve as a lightweight and efficient password enhancer for applications in IoT devices.

6.1 Preliminaries on Password Application

For the human-generated passwords, the problem caused by lacking entropy is severe. People are notoriously poor at achieving sufficient entropy to produce satisfactory passwords. The lack of entropy indicates the low-security level of the human-generated passwords which could cause problems in the authentication procedure. According to one study involving half a million users, the average entropy of passwords was estimated as 40.54 bits [21]. It indicates 5 symbols (8 bits entropy represents one symbol in ASCII table) are in the password which is far away from the requirements of the password as 8 symbols.

Moreover, there exist some patterns in human-generated passwords which could be easily learned by the malicious attackers. Users rarely make full use of larger character sets in forming passwords. For example, hacking results obtained from a MySpace scheme in 2006 revealed 34,000 passwords, of which only 8.3 percent used mixed case, numbers, and symbols. As a result, an eight-character human-selected password without uppercase letters and non-alphabetic characters is estimated to have 18 bits of entropy which is far from the randomly selected passwords as 64 bits (8 symbols and 8 bits per symbol). The above Results demonstrate that from the security perspective, human generated passwords are far away from the borderline of password security. However, the user-generated password has its advantages as easy to use and easy to remember. Since there exists the entropy gap between the human-generated password and the entropy requirement of the password, our entropy pump is applicable to solve this problem by offering entropy improvement for the human-generated password.

6.2 Evaluation on Password Dataset

6.2.1 Password Test Sets

The password sets we used in this article are collected from the RockYou password list [22]. The RockYou list was originally obtained by a hacker who utilized a SQL injection attack against the rockyou.com website, and then later posted the passwords online. RockYou provided applications for numerous social networking sites such as Facebook, MySpace, and Friendster. The actual list itself contained over 32 million passwords. Due to the list's enormous size, we select typical sub-sets labeled as "the most common XX passwords", which "XX" stands for the number of passwords in the set. To make the password test implementable on the PUF array structure, we choose the first 32 bits as 4 characters in each password for testing.

6.2.2 Experimental Results

We first test the most common passwords sets from the RockYou list and results are as we expected showing that the human-generated passwords have low entropy as in Fig. 7. The group of bars drawn in the leftmost represents the entropy of "the most 500 common passwords" case which is the lowest, and the entropy has a trend to increase with enlarging the size. The middle bar in each group represents the output entropy which increases from the original entropy (around 6 bits). Moreover, by applying the exclusive-or method, the entropy increases to 7.5 bits which is much closer to the full entropy as 8 bits. These results demonstrate that our entropy pump could certainly improve the low entropy of the most common human generated passwords sets.

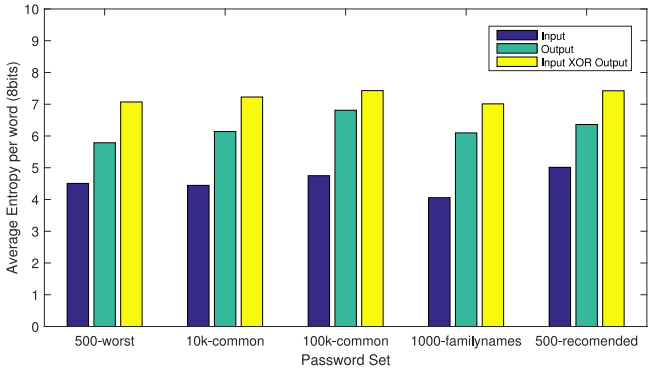

Fig. 7. The entropy results for different password sets.

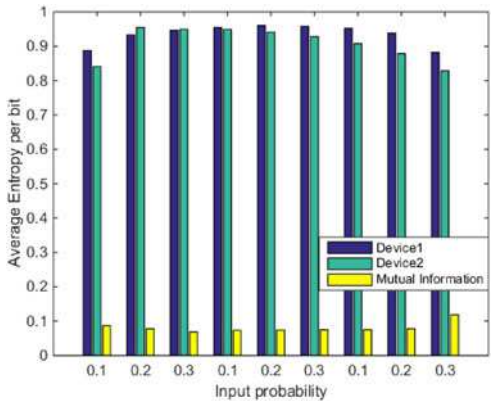

Fig. 8. Mutual information result of two devices.

We also test the "most 1000 common English family name" set in contrast with the password sets. In this case, the original entropy is lower than the human generated password set because all the characters for family names are alphabet letters. However, the output entropy, as shown in Fig. 7, has increased to almost the same level as the passwords sets. This result indeed demonstrates that our PUF pump is effective for even worse inputs.

In addition, we also test the recommended human generated password in the RockYou list and the results are drawn on the right most of Fig. 7. The good human-generated password is consist of numbers intersecting with letters, so the input entropy is around 5 bits which is much higher than the other cases but still has 3-bit gaps to the full entropy. However, by being processed of our entropy enhancer, the output entropy increased to 7.8 bits which is very close to full entropy.

6.3 Mutual Information Evaluation

To prove that our proposed PUF holds uniqueness implemented in different devices, we test the mutual information (MI) of the outputs from two devices. In the information theory, mutual information of two random variables is a measurement of the mutual dependence between the two variables. Mutual information can be equivalently expressed with entropy as,

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \\ &= H(X) + H(Y) - H(X,Y), \end{aligned} \tag{14}$$

where $H(X)$ and $H(Y)$ are the marginal entropy, $H(X|Y)$ and $H(Y|X)$ are the conditional entropy, and $H(X,Y)$ is the joint entropy of X and Y.

In the password application, the ideal mutual information of two devices should be as small as close to 0. That is because if two customers occasionally generate the same password, the final passwords from the entropy enhancer should not be the same. This guarantees the attacker cannot obtain the passwords of the other machines if he knows the passwords generated from one machine. Otherwise, if we replace our PUF based enhancer by the cryptography algorithms without any other random sources, e.g., the salt value, different devices would result in the same output. Because those functions are identical on different devices, this scheme cannot resist attacks as pre-computed attacks. To solve this authentication problem, the device implemented with hash functions needs to add extra random generation functions to concatenate a random number with the password. However, in our scheme, the random function is embedded in the PUF structure, thus we can save the cost both in hardware and time.

The results of the mutual information of two devices are shown in Fig. 8. The output entropy of Device 1 and Device 2 are all above 0.85 for 9 different input cases. On the contrary, the mutual information results of two devices are all less than 0.1. Results demonstrate that by knowing the output of one device, the attacker cannot get useful information on the other device.

7 Related Work

From the above discussion, our proposed entropy pump has been demonstrated in enhancing the entropy of input source using the intrinsic randomness of hardware PUF. It could be used as a booster for the random source or a low-cost replacement for distiller functions, making it a promising hardware security primitive for IoT applications. In this section, we survey the most commonly used random number generators (RNG) and discuss how our proposed entropy pump can be used to improve them.

7.1 Random Number Generators

A true random number generator (TRNG) uses a non-deterministic source, e.g., thermal noise, circuit noise or quantum effects, as the entropy source to produce randomness. In order to guarantee the randomness for TRNG, some post-processing functions, such as the entropy distillation [5], are used to overcome the weakness in the entropy source.

There are some popular TRNGs based on semiconductor chips. Electronic noises and time jitter are usually the most common stochastic phenomenon that are suitable for the integration in embedded systems as chip-card controllers [23], [24]. Chaos theory based chaotic behaviors have provided an alternative and qualitatively different type of random number generators [25]. Besides, the initial SRAM states can also be used as a source of TRNG for identifying fingerprints. The state randomness is unpredictably scattered throughout the SRAM and must be collected by an entropy extractor [26]. Transition effect ring oscillators (TERO) based TRNGs have been proposed

to obtain randomness from the oscillatory metastability in circuits. More specically, the randomness of TERO element is from a combination of transient oscillatory metastability and the behavior of bistable flip-flops [23], [27], [28]. A post-processing procedure is normally associated with these TRNGs for entropy distillation by utilizing the diffusion and confusion properties of cryptographic functions [5], [23], [29].

Quantum random-number generators (QRNGs) are believed to be a good entropy source to generate information theoretically provable random numbers in principle [30]. But in practice, unfortunately, the quantum randomness is inevitably mixed with classical noises. To distill this quantum randomness, one needs to quantify the randomness of the source and apply a randomness extractors [31]. The extractors would add additional cost to the already expensive QRNG, making it not applicable for IoT applications because of their limited resources. However, randomness beacons have been reported where the entropy is generated by the quantum source [32].

Random numbers in most applications are from pseudorandom number generators which use deterministic processes to generate a series of outputs from an initial seed state (e.g., from the TRNG). Because the output is purely a function, which describes the PRNG, of the seed value, the actual entropy of the output can never exceed the entropy of the seed. This limits the entropy of PRNGs and hence the security level it can provide in emerging applications [33], [34].

7.2 Cost of Random Number Generators

The cost of the entropy generation can be evaluated based on many criteria and is of great importance for many applications, in particular those resource constrained embedded systems and IoT devices. There are two major cost related to random number generators: the cost to collect the randomness and the cost for the post-processing distiller. As we discussed earlier, entropy could be obtained from different entropy sources. In general, the equipment required for TRNGs to capture physical randomness is very expensive. For instance, we have learned that quantum source may generate good randomness, but capturing the quantum effects can cost thousands of times higher than the cost of collecting the randomness from the noise that could be achieved in small circuits.

In addition to the equipment cost for TRNGs, another negligible cost is related to the post-possessing procedure called distiller. Hardware based random number generators can feature a very high throughput, however, even when well-designed, the produced bit streams usually show a certain level of correlation due to bandwidth limitation, fabrication tolerances, aging and temperature drifts, and deterministic disturbances. To address this problem, a common procedure to remove statistical imperfections in the output bit stream is to process the sequence with a carefully designed correcting algorithm. The algorithm uses a high speed near-random input stream to generate a lower speed bit stream with increased statistical quality, distilling the entropy contained in the input sequence [23]. Most of random number generators need the distiller compression algorithm implemented either in software as the hash function or in hardware used long shift registers. For both the software and hardware implementations, the time delay may cause problems as the reaction of the random number generator will be delayed by this process. Distilling codes and circuitry are expensive in terms of the number of raw bits and silicon resources required. Moreover, the post-processing procedure may disclosure the randomness to malicious attackers.

7.3 Relationship of Our Work with the Previous Arts

Silicon PUFs, including RO PUF, have been used to create the seed for random number generators [24], [27], [28]. It is important to differentiate our PUF entropy pump from those early works. Our goal is to improve the entropy of random numbers, not to create random numbers from scratch. So the entropy pump can be considered as a low cost auxiliary to random number generators.

For TRNGs, our proposed PUF based entropy enhancer plays a similar role of the post-processing distiller in enhancing the entropy of output from TRNG. The key advantage of our approach is its lower hardware cost compared to traditional cryptographic functions used in the distillers. For example, our PUF based entropy pump requires only 569 slices in the FPGA implementation, while popular cryptographic algorithms would cost 1000-2000 slices. As the data from both simulation and FPGA implementation shows, our entropy pump can enhance entropy to a level very close to the full entropy. Therefore, it can be considered as a replacement of the expensive distiller for devices constrained by resources.

The TERO based designs share some similarities with our proposed entropy pump, although they are two completely different structures designed for different applications. The TERO captures the transition instability to generate random numbers, however, in our cases, we utilize the intrinsic delay difference in boosting the entropy from a given source. TERO can be considered as an entropy source where the randomness is derived from the intrinsic jitters. In that aspect, it will be interesting to investigate whether our entropy pump can improve the entropy of TERO.

For QRNG, recall that the randomness extracted from the quantum effects cannot be used directly because of noises. It is possible to use our proposed entropy enhancer to distill such noise and make the bit stream generated by the quantum effects usable for applications in IoT devices. On the other hand, for random number generators that do not need further entropy improvement such as the NIST randomness beacon [32], the proposed PUF entropy pump can be used for security purpose. For instance, when a QRNG is provided by an untrusted manufacturer, the random output of the randomness beacon can be pre-generated and recorded such that the manufacturer can always predict the output. By our PUF entropy pump, the untrusted output will be "encrypted" and the manufacturer will not be able to predict it unless the entropy pump is compromised. Notice that our simulation and FPGA implementation data has demonstrated that the PUF entropy pump will be cost significant entropy loss when the input entropy is high. Therefore, this approach will add security to the high entropy randomness beacon.

8 Conclusions

The strength of cryptographic keys plays a central role in security. However, high entropy sources normally come with a high cost and traditional entropy enhancement methods are also expensive. In the article, we propose a novel approach to boost the entropy of keys generated from low entropy sources by utilizing intrinsic randomness in the physical devices. More specifically, we use the low entropy key as the configuration bits to configure a recently developed highly flexible ring oscillator PUF. This will integrate the unpredictable silicon fabrication variation with the low entropy key to produce potentially high entropy outputs, which has been validated by both simulation and FPGA implementation. The low-cost feature of RO PUF makes our proposed system a perfect hardware security primitive for many IoT applications where the devices have limited resources and the applications do not require high-security level. We demonstrate this with the implementation of a PUF-based password system. Our experiments show that this method can enhance the entropy from about 5 bits per character to 7.5 bits per character, a 50 percent improvement.

Acknowledgments

Qian Wang and Gang Qu are supported in part by AFOSR MURI under award number FA9550-14-1-0351.

References

[1] A. Vassilev and T. A. Hall, "The importance of entropy to information security," *Comput.*, vol. 47, no. 2, pp. 78–81, 2014.
[2] E. B. Barker, J. M. Kelsey, "Recommendation for random number generation using deterministic random bit generators (revised)," US Department of Commerce, Technology Administration, National Institute of Standards and Technology, Computer Security Division, Information Technology Laboratory, 2007.
[3] H. Xu, S. Zhou, L. Xiao, H. Wang, S. Li, and Q. Yuan, "Fabrication of a nitrogen-doped graphene quantum dot from mof-derived porous carbon and its application for highly selective fluorescence detection of fe 3+," *J. Mater. Chemistry C*, vol. 3, no. 2, pp. 291–297, 2015.
[4] Y. Dodis and A. Smith, "Entropic security and the encryption of high entropy messages," in *Proc. Theory Cryptography Conf.*, 2005, pp. 556–577.
[5] X. Ma, F. Xu, H. Xu, X. Tan, B. Qi, and H.-K. Lo, "Postprocessing for quantum random-number generators: Entropy evaluation and randomness extraction," *Phys. Rev. A*, vol. 87, no. 6, 2013, Art. no. 062327.
[6] B. Gassend, D. Clarke, M. Van Dijk, and S. Devadas, "Silicon physical random functions," in *Proc. 9th ACM Conf. Comput. Commun. Secur.*, 2002, pp. 148–160.
[7] G. E. Suh, C. W. O'Donnell, I. Sachdev, and S. Devadas, "Design and implementation of the aegis single-chip secure processor using physical random functions," in *Proc. ACM SIGARCH Comput. Archit. News*, vol. 33, no. 2, 2005, pp. 25–36.
[8] O. Kara and A. Atalay, "Preimages of hash functions through rainbow tables," in *Proc. 24th Int. Symp. Comput. Inf. Sci.*, 2009, pp. 304–309.
[9] A. Maiti and P. Schaumont, "Improving the quality of a physical unclonable function using configurable ring oscillators," in *Proc. Int. Conf. Field Programmable Logic Appl.*, 2009, pp. 703–707.
[10] M. Gao, K. Lai, and G. Qu, "A highly flexible ring oscillator puf," in *Proc. 51st ACM/EDAC/IEEE Design Autom. Conf.*, 2014, pp. 1–6.
[11] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE Mobile Comput Commun. Rev.*, vol. 5, no. 1, pp. 3–55, 2001.
[12] S. Katzenbeisser, Ü. Kocabaş, V. Rožić, A.-R. Sadeghi, I. Verbauwhede, and C. Wachsmann, "Pufs: Myth, fact or busted? a security evaluation of physically unclonable functions (pufs) cast in silicon," in *Proc. Int. Workshop Cryptographic Hardware Embedded Syst.*, 2012, pp. 283–301.
[13] R. Van Den Berg , B. Skoric, and V. van der Leest, "Bias-based modeling and entropy analysis of pufs," in *Proc. 3rd Int. Workshop Trustworthy Embedded Dev.*, 2013, pp. 13–20.
[14] J. Delvaux, D. Gu, and I. Verbauwhede, "Upper bounds on the min-entropy of ro sum, arbiter, feed-forward arbiter, and s-arbro pufs," in *Proc. IEEE Asian Hardware-Oriented Secur. Trust*, 2016, pp. 1–6.
[15] C. E. Shannon, "Communication theory of secrecy systems," *Bell Syst. Tech. J.*, vol. 28, no. 4, pp. 656–715, 1949.
[16] M.-D. Yu, M. Hiller, J. Delvaux, R. Sowell, S. Devadas, and I. Verbauwhede, "A lockdown technique to prevent machine learning on pufs for lightweight authentication," vol. 2, no. 3, pp. 146–159, Jul./Sep. 2016.
[17] Q. Wang, M. Gao, and G. Qu, "A machine learning attack resistant dual-mode puf," in *Proc. Great Lakes Symp. VLSI*, 2018, pp. 177–182.
[18] M. S. Farash and M. A. Attari, "An efficient and provably secure three-party password-based authenticated key exchange protocol based on chebyshev chaotic maps," *Nonlinear Dyn.*, vol. 77, no. 1/2, pp. 399–411, 2014.
[19] J. Ding, S. Alsayigh, J. Lancrenon, R. Saraswathy, and M. Snook, "Provably secure password authenticated key exchange based on rlwe for the post-quantum world," in *Proc. Cryptographers Track RSA Conf.*, 2017, pp. 183–204.
[20] M. H. Almeshekah, C. N. Gutierrez, M. J. Atallah, and E. H. Spafford, "Ersatzpasswords: Ending password cracking and detecting password leakage," in *Proc. 31st Annu. Comput. Secur. Appl. Conf.*, 2015, pp. 311–320.
[21] D. Florencio and C. Herley, "A large-scale study of web password habits," in *Proc. 16th Int. Conf. World Wide Web*, 2007, pp. 657–666.
[22] Y. Zhang, F. Monrose, and M. K. Reiter, "The security of modern password expiration: An algorithmic framework and empirical analysis," in *Proc. 17th ACM Conf. Comput. Commun. Secur.*, 2010, pp. 176–186.
[23] M. Varchola and M. Drutarovskỳ, "New high entropy element for fpga based true random number generators," in *Proc. Int. Workshop Cryptographic Hardware Embedded Syst.*, 2010, vol. 6225, pp. 351–365.
[24] M. Bucci, L. Germani, R. Luzzi, A. Trifiletti, and M. Varanonuovo, "A high-speed oscillator-based truly random number source for cryptographic applications on a smart card ic," *IEEE Trans. Comput.*, vol. 52, no. 4, pp. 403–409, Apr. 2003.
[25] T. Stojanovski and L. Kocarev, "Chaos-based random number generators-part i: Analysis [cryptography]," *IEEE Trans. Circuits Syst. I: Fund. Theory Appl.*, vol. 48, no. 3, pp. 281–288, Mar. 2001.
[26] D. E. Holcomb, W. P. Burleson, K. Fu, et al., "Initial sram state as a fingerprint and source of true random numbers for rfid tags," in *Proc. Conf. RFID Secur.*, 2007, vol. 7, Art. no. 2.
[27] M. Majzoobi, F. Koushanfar, and S. Devadas, "Fpga-based true random number generation using circuit metastability with adaptive feedback control," in *Proc. Int. Workshop Cryptographic Hardware Embedded Syst.*, 2011, pp. 17–32.
[28] P. Z. Wieczorek and K. Golofit, "Dual-metastability time-competitive true random number generator," *IEEE Trans. Circuits Syst. I: Regular Papers*, vol. 61, no. 1, pp. 134–145, 2014.
[29] V. Fischer, "A closer look at security in random number generators design," in *Proc. Int. Workshop Constructive Side-Channel Anal. Secure Des.*, 2012, pp. 167–182.
[30] T. Jennewein, U. Achleitner, G. Weihs, H. Weinfurter, and A. Zeilinger, "A fast and compact quantum random number generator," *Rev. Sci. Instruments*, vol. 71, no. 4, pp. 1675–1680, 2000.
[31] J. D. Golic, "New methods for digital generation and postprocessing of random data," *IEEE Trans. Comput.*, vol. 55, no. 10, pp. 1217–1229, Oct. 2006.
[32] NIST, "Nist randomness beacon," Sep. 2011 (updated Dec. 6, 2017). [Online]. Available: https://www.nist.gov/programs-projects/nist-randomness-beacon
[33] J. Viega and G. R. McGraw, *Building Secure Software: How to Avoid Security Problems the Right Way*. London, U.K.: Pearson Education, 2001.
[34] D. Kaplan, S. Kedmi, R. Hay, and A. Dayan, "Attacking the linux prng on android: Weaknesses in seeding of entropic pools and low boot-time entropy," in *Proc. 8th USENIX Conf. Offensive Technol.*, 2014, p. 14.

Qian Wang received the MS degree in electrical engineering from Tsinghua University, China, in 2014. She is working toward the PhD degree in the Maryland Embedded Systems and Hardware Security Lab, University of Maryland, College Park. Her research interests include embedded system and hardware security such as side channel attacks and PUF-related applications.

Gang Qu (SM'07) received the BS and MS degrees in mathematics from the University of Science and Technology of China, Hefei, China, in 1992 and 1994, respectively, and the PhD degree in computer science from the University of California at Los Angeles, in 2000. He joined the University of Maryland at College Park, where he is currently a professor with the Department of Electrical and Computer Engineering and the Institute for Systems Research. He is the director of the Maryland Embedded Systems and Hardware Security Laboratory, College Park. His primary research interests include embedded systems and very large-scale integration (VLSI) computer-aided design (CAD) with a focus on low-power system design and hardware-related security and trust.

PUF-PassSE: A PUF based Password Strength Enhancer for IoT Applications

Qian Wang, Mingze Gao, Gang Qu
Department of Electrical and Computer Engineering
University of Maryland, College Park, MD USA
E-mail: {qwang126, mgao1, gangqu}@umd.edu

Abstract

Authentications and encryptions are urgently needed by the growing number of hardware devices for security applications of the Internet of Things (IoT). However, the existing cryptographic solutions may not be applicable to IoT devices because of the strict timing and power requirements of the devices. As a result, Physical Unclonable Function (PUF) is a promising hardware primitive to provide security in existing and future IoT applications due to it is lower hardware cost and energy efficient. In this paper, we propose an innovated application of PUF as an entropy pump to improve the entropy. Furthermore, we apply the PUF-based entropy pump in a password enhancement application for embedded IoT devices. We confirmed that an overall 48% significant improvement on the entropy of the human-generated passwords. The proposed design is demonstrated on the Nexys 4 DDR FPGA board with low hardware overhead, which ensures the feasibility for IoT applications.

Keywords

PUF, Entropy source, Password, IoT

1. Introduction

The arrival of the IoT era offers a unique opportunity for hardware-based security. The primitive of the IoT is that the devices (e.g., microcontrollers, RFID receivers and variety of sensors) are connected in the network and are able to collect and exchange data. And the IoT network allows objects to be sensed and controlled remotely across existing network infrastructure. However, those applications generate emergent security concerns on the IoT network and make security one of the most challenging tasks. Since IoT devices are constrained with limited CPU, memory and battery power, the existing computational intensive cryptographic algorithms and protocols cannot be efficiently implemented on such devices, leaving them vulnerable. As a kind of lightweight hardware security primitives, PUF has the properties of low-power, small area, unpredictability, and unclonability. Thus, PUF (Physical Unclonable Function) is considered to be a promising primitive who is secure and efficient to meet the requirements of IoT-based applications [1][2]. PUF is a hardware implemented one-way function utilized the variance of fabrication which cannot be predictable [3]. Despite for the already existing PUF applications for device authentications [4][5], we propose a new application as a PUF-based password strength enhancer (PUF-passSE) to improve the entropy of passwords for IoT devices.

For today's Internet, websites authenticate users by requiring a password through the Secure Sockets Layer (SSL) protocol. The IoT scale authentication also borrows this idea and most of the devices in the network require valid passwords before executing program or sharing data. Unfortunately, as bad as passwords have been for Internet-scale authentications, they are even worse for the IoT. One of the main reasons is because that the resource restriction of IoT device limiting the length of passwords. Since the Id/password pairs are relatively lightweight, managing them in high quality is not practical. Another concern is that the security credential of the password cannot be guaranteed because of the reported malicious attacks to break passwords [6]. For example, the adversary who steals the list of hashed passwords can use brute-force to pre-compute the rainbow table thus discovering a password stored for a user [7, 8]. For the IoT-scale devices, the threat model is similar as the brute force attack on Internet. The adversary can precompute the valid passwords and use it for device authentication or user authentication. However, the valid passwords are usually shorter for the IoT-scale, thus making it even easier for pre-computing. Besides, using the human-generated passwords for authentication makes the precarious credential even worse as the intrinsic lack of randomness. On the contrary, fulfilling minimum length and character type requirements while attempting to create something memorable can become an arduous task, leaving the users frustrated and confused. How to filling the gap between the human-generated passwords and the 'good' passwords becomes a percussive problem that must be solved to guarantee security for IoT applications.

After discussing the vulnerabilities related to passwords, it naturally arouses one question: how can we evaluate the property of password? Here, we use the entropy in information theory as the metric to evaluate the strength of password. After that, the next critical question is how can we increase the entropy of passwords. In this paper, we propose a PUF based entropy enhancer which takes use of the PUF's intrinsic unclonability and randomness. The PUF-passSE is designed to be applied in enhancing the password strength for lightweight IoT authentications. Specifically, it could achieve the following purposes for IoT applications.

1) First of all, the embedded randomness of the PUF could serve as a security enhancer which could pump the entropy source, thus strengthen the password with higher entropy.

2) Secondly, the physical properties also assemble the PUF with uniqueness between different devices which could be utilized to resist the pre-compute attacks for passwords. It

could achieve in making password cracking impossible without physical access to the devices. For example, if the attacker pre-computes the passwords on one device, the pre-computed table will become useless for other devices, which significantly hinders the attacker's ability to implement attacks on a large number of devices of the same type.

3) At last, the PUF-passSE could also assist in the password authentication procedure with human interfaces. The user just needs to remember his own password, while the PUF-passSE can process the original insecure password to generate a new password with higher entropy. Moreover, the hardware-module is easily to be embedded into any portable devices.

In the remainder of this paper, we will make this high-level intuition precise in the following way: first, we introduce the backgrounds on the password related threats and define the notation of password strength and entropy of the password in Section 2. Then, in Section 3, we describe the design of RO PUF. In Section 4, we give proof on the improvement of entropy. Next, we introduce the system-level implementation on FPGA board in Section 5. We test the entropy on the password dataset to validate our claim in Section 6 and finally concludes the paper.

2. Background and motivation

2.1. Human generated passwords and problems

Passwords are the most dominant form of authentication and will remain in use for many users despite their weakness to malicious threats. Naturally, people are notoriously poor to generate satisfactory passwords even with the guidance from some password policies. According to one study involving half a million users, the average entropy of passwords is around 40.54 bits [10]. The number indicates that human-generated password on average contains 5 characters (8 bits entropy for a character) which is less than the password policy which requires password should have a minimum length of 8. Moreover, people inevitably generate passwords with some regular patterns which are easy to be discovered by the attackers. For example, in one analysis of over 3 million eight-character passwords, the letter ‘e’ was used over 1.5 million times (50% of passwords), while the letter ‘f’ was used only 250,000 times (8.3%) [10]. Users rarely make use of the full character set in forming passwords. For example, hacking results obtained from a MySpace scheme in 2006 revealed 34,000 passwords, of which only 8.3% used mixed case, numbers, and symbols [11]. Apparently, those drawbacks of human-generated passwords would cause serious security issues in the authentication procedure. An eight-character human-selected password without uppercase letters and non-alphabetic characters is estimated to have 18 bits of entropy which is worse than the average as 40.54 bits, and far from the requirements as 64 bits. The 8-character lowercase alphanumeric passwords case would easily tractable using a personal computer with time estimated as 5 hours. Therefore, results from the above studies prove that the human-generated password lacks entropy and is vulnerable to attacks. However, users continue to use poor passwords because of the convenience of those passwords. Thus, it is necessary to help people in generating more secure passwords to fulfil the authentication procedure from malicious threats. Despite the instructive password policies, we propose a hardware assist method based on PUF to enhance the password security and overcome the shortcomings of the human generated passwords.

2.2. Password strength

Usually, password consists of a set of symbols in which each symbol is equally likely to be selected from a symbol set (e.g., the ASCII character set). To measure how unpredictable the password is, entropy is brought up as a criterion. The idea of information entropy was an approach to measure the amount of information that is unknown to random variables [9]. Most often this randomness or information is expressed using the following equation as the definition of entropy:

$$H(x) = \sum_{i=0}^{n} P(x_i) \cdot \log_2 x_i \qquad (1)$$

Similarly, entropy can be used to measure the strength of passwords which depends on the actual randomness in selecting symbols. Password entropy predicts how difficult a given password would be to crack through guessing, brute force cracking, dictionary attacks or other common methods. For example, for passwords generated by a process that selects a set of symbols of length, L, from N possible symbols, the number of possible passwords can be acquired by raising the number of symbols to the power L, i.e. N^L. The strength of a random password as measured by entropy is conduct base-2 logarithm on the number of possible combinations. Obviously, increasing either L or N will strengthen the password by enhancing the entropy. Therefore, the information entropy of passwords, H, is given by the formula:

$$H = \log_2 N^L = L \log_2 N = L \frac{\log N}{\log 2} \qquad (2)$$

Password entropy is usually expressed in terms of bits. A password that is already known has 0 entropy; one that would be guessed on the first attempt half the time would have 1 bit of entropy. A password's entropy can be calculated by finding the entropy per character, which is a log base 2 of the number of characters in the character set used, multiplied by the number of characters in the password itself.

From the definition, we could find out that the full strength of entropy when using the ASCII character set (numerals, mixed case letters, and special characters) is achieved only if each character in the password is chosen with equal probability from that set. From this claim, we could find out that capitalizing a letter or adding numbers or special character to a password will not make the password more secure. Even worse is that if the numbers and special characters are added in a predictable way, saying at the beginning and end of the password, it will decrease the password strength compared to when all letters are random selected. As a result, instead of padding characters, it is

necessary to find a solution to enhance the security level of the passwords.

2.3. Related work

A number of cryptographic functions have been used in computer systems to protect passwords. The motivation to develop additional algorithms is to make the cracking process of stolen passwords to become resource intensive. However, most of those schemes are designed for Internet-scale applications, without considering the additional cost for these underlying functions. However, for most proposals are designed for Internet-scale applications which either need post-quantum cryptography or expensive cryptography algorithms (i.e., Diffie-Hellman protocols) [16], [17]. There are the other attempts trying to use a hardware security module (HSM) at the authentication server to prevent off-site password discovery [18]. While their scheme is designed for the Internet scale as they try to protect the stored user passwords in the server from cracking by the adversary. Our proposed entropy pump is designed for light-weight application in IoT devices, where we focus on enhancing the entropy of the human generated passwords from the user side, not how the passwords can be stored securely on the server side.

3. Design of PUF-passSE

Due to the fact that process variation is unclonable, the input-output mapping of an individual PUF is deterministic but unpredictable. We utilize this property to build our password entropy enhancer.

Our proposed PUF-passSE is built on a new design type of PUF called the configurable RO PUF. The initial idea for configurable PUF is derived from the structure proposed by Gao et al. [12] [13]. The original RO PUF consists of an odd number of inverters and connected from head to tail. A multiplexer is added after each inverter which controls whether selecting this inverter into the final oscillator ring or not. The selection signal of the multiplexer is defined as the configurable input or configurable bit.

Fig.1. shows the configurable RO PUF design for the PUF-passSE. We make some significant modifications on the original design to realize the new application as the password enhancer. First, we choose the number of configurable bits as 8 because passwords are represented by a character and each character could be mapped to an 8-bit binary value. In this case, we place 8 inverters in the ring. However, a circular ring composed of an even number of inverters cannot oscillate, because the output of the last inverter is always the same as the input of the first inverter. To make the ring oscillate, we add a backup inverter in the ring, as marked in dash line in Fig.1. The backup inverter works as follows. If the 8-bit configurable input has odd number of '1's, which could generate the oscillation, we just make the selection bit of the backup inverter '0'. Otherwise, we will connect the backup inverter in the ring by setting the configurable bit to '1'. A simple method to generate the configuration bit for the backup inverter in hardware is by bit-wise exclusive-or the 8-bit input.

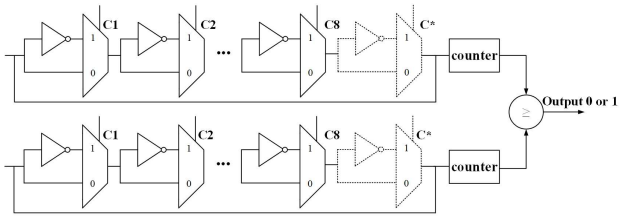

Fig. 1. Schematic of the RO PUF design, 8 inverters with a back-up inverter (drawn in dash) make up a single ring, the output is generated by comparing the time delay of two parallel rings.

4. PUF model analysis

As we stated before, PUF-passSE design has 8 stages for each ring, and the configurable input is defined as challenge *C*. The backup input bit is the parity bit of the challenge which could be defined by the exclusive-or function. The PUF function would work on the concatenation of the input and generate one-bit response for each ring. The PUF function could be represented as:

$$R = puf(C \| \oplus_{i=1}^{n} C_i) \qquad (3)$$

Here, we will give some theoretical proofs of why the PUF based enhancer could enhance the entropy of the password. We will start with the definition of one-way function and the construction of pseudorandom permutations based on this [14].

Step 1, one-way function: PUF is believed to be a one-way function following the definition as it is easy to compute but hard to invert. With the more specific definition as

Definition 1: PUF is a function $f:\{0,1\}^* \rightarrow \{0,1\}^*$ is one-way if the following two conditions hold:

1. Easy to compute: There exists a polynomial-time algorithm M_f computing f, that is $Mf(x) = f(x)$ for all x.
2. Hard to invert: It cannot be converted in a polynomial time.

Step 2, from one-way function to one-way permutation: In our PUF-pass design we make the output is consistent with the input length as both of them are 32-bit. It is indeed constructing the one-way permutations. A one-way permutation is a one-way function with additional structural properties. We say a function is length-preserving if $|f(x)|=|x|$ for all x. A one-way function that is length-preserving and one-to-one is called a one-way permutation.

Step 3, Hard-core predicates:

By definition, a one-way function is hard to invert. But it is not always the case that nothing about x can be determined in polynomial time from $f(x)$. For more specific applications, we need to find some information about x that is hidden by $f(x)$. This motivates the notion of a hard-core predicate.

The definition of hardcore predicate hc: $\{0,1\} \rightarrow \{0,1\}$ of a function f has the property that hc(x) is hard to compute with probability significantly better than ½ given $f(x)$. We stress that hc(x) is efficiently computable given x. the definition requires that hc(x) is hard to compute given $f(x)$.

One of the typical hard-core predicts used in cryptography is defined as $\mathrm{hc}(x) = \oplus_{i=1}^{n} x_i$ where $x_1 \ldots x_n$ denote the bit of x. For our PUF function, we can write it as $f(\mathrm{hc}(x))$. And it is clear to show that hc is a hard-core predicate of f. If f cannot be inverted, then $f(x)$ must hide at least one of the bits x_i of its preimage x, which would seem to imply that the exclusive-or of all the bits x is hard to compute.

Step 4, from one-way functions to pseudorandom permutations:

We know that the one-way function can construct pseudorandom permutations. Here we will prove how the one-way function PUF to fulfill this. First, we have shown that the hard-core predicts of the PUF one-way function.

Since hc is to be a hard-core predicate of f, then the PUF function can be written as a function directly on the PUF function as $G(s) = f(\mathrm{hc}(s)\|s)$, where G is a pseudorandom generator. This is intuitively why G, as defined in the theorem, constitutes a pseudorandom generator, not first that the initial n bits of the output $f(s)$ are truly uniformly distributed when s is uniformly distributed. Next, the fact that hc is a hard-core predicate of f means that $\mathrm{hc}(s)$ 'looks random'.

5. System-level FPGA implementation

We implement the PUF-passSE design on the Xilinx Artix-7 FPGA in the Nexys4-DDR board and we also build a graphical user interface (GUI) for the PUF-based password strength enhancement application demo.

5.1. RO placement

We place in total of 32 groups of RO instances on the FPGA chip. The input of each group is also 32-bit which could be translated from 4 characters. The input 32-bit will be sliced into 8-bit for each slice and the 8-bit configuration input is for the 9-stage RO. Note that the last bit is the parity to generate the oscillation. Thus, the 32-bit input would be grouped into 4 sub-groups and each sub-group has the same structure as depicted in Fig.1. To make the output consistent with the input, the responses from the 4 sub-group will exclusive-or to form a 1-bit final output. Thus, one pair of ROs will generate 1-bit output and the combined output is 32-bit. If the password length is longer than 32-bit, we address this as follows. We divide the input into chunks of length of 32 and then feed each chunk to the 32 groups of RO. After that, if the left part is less than 32-bit, we will just initialize partially the corresponding sub-group in terms of 8-bit. When placing the ROs, we use the relative location constraints (RLOC) from the Xilinx EDA tools to ensure the delay difference of each ring is generated by the variance of the die layout.

5.2. System implementation

We build a serial port communication from the FPGA board to PC using the universal asynchronous receiver/transmitter forms. To make the communication compatible with the 32-bit RO PUF, the communication width is set to be 32-bit per frame and any length passwords will be split into 32-bit (4 characters). A graphical user interface is shown in Fig.2. The input string is the user-generated password and it is first converted into ASCII code. The actual input of the RO PUF is the binary value converted from the ASCII code. The output is translated to the corresponding enhanced password from the binary code shown at the bottom of Fig.2.

Both the input and output passwords are presented in Fig.2 to make a comparison. In addition, we also display the corresponding hex and binary values for reference. But note that in real applications the intermediate values are not exposed for security consideration. That is to say, the user only needs to remember his personal password, no matter how simple it is. The embedded PUF-passSE will convert it into a strong password and use that one in the following authentication.

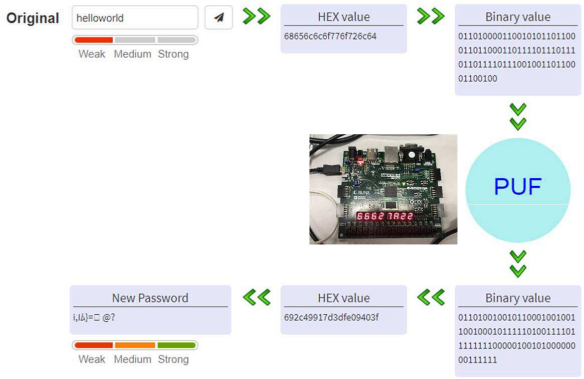

Fig. 2. A typical demo of the PUF-passSE System, the input is the original password which is labled as 'weak' by the password security checker, while after processing by the PUF model, the newly generated password is labeled as 'strong'.

6. Results and analysis

To evaluate the applicability of our PUF-passSE on real-world passwords, we assess its performance in terms of entropy enhancement for the password dataset acquired from the online servers. Moreover, we also test the results on different FPGA boards and evaluate the mutual information between different devices.

6.1. Bit entropy result

The concept of PUF entropy is to evaluate the randomness and unpredictability of the PUF readouts. The higher the entropy means the output/password is more random. It also indicates that attacker faces more difficulties in guessing or breaking the output/password. We begin verifying the performance of the PUF based entropy enhancer by calculating the bit entropy enhancement. First, we generate the input random bit-stream obeys Bernoulli distribution $B\sim(n,p)$ with different probabilities (p). For example, when the probability equals 0.1, it indicates the frequency of '1' occurred in the bit-stream with proportion as 10%. The entropy, in this case, is low because of the majority of the bits in the stream is '0', thus lacking

randomness. When p=0.5, it means '1' and '0' occur in the bit-stream with the same probability. It shows the most random case for Bernoulli distribution and the entropy, in this case, is equals the full entropy 1. As for p=0.9, in which '1' occurs 90% of the bit-stream, the entropy is also low. We evaluate the average bit entropy of the input stream, output stream and as well as the bit exclusive-or result of the two, shown in the Fig.3. First, from the figure we see a clear increase from the input entropy to the output entropy. However, when the input entropy is already high which is close to full entropy, passing by the PUF, the corresponding output entropy decreases a bit, i.e, 0.05. The decrease is because PUF output is not ideal random in implementation. Some variance in the hardware may add bias to the PUF output. To overcome this limitation, we propose to exclusive-or the output with the initial input. This solves the problem when the input entropy is already near the full entropy. In Fig.3, it is evident to see the exclusive-or entropy H_{xor} is higher than the original output entropy H_{out}. This method guarantees that the enhancement still holds when the input entropy is already high. In summary, from the binary entropy result, we demonstrate that the PUF-passSE can enhance the input entropy of bit-streams.

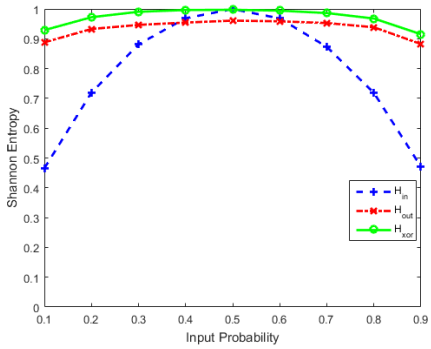

Fig. 3. Bit entropy result for 9 different cases

6.2. Result of password dataset

After confirming the increase of entropy for generated bit streams, we conducted the following experiment to further validate the enhancement of the entropy on passwords dataset. We use the passwords from the RockYou password set [15] to validate our PUF-passSE. The RockYou list was originally obtained by an attacker who utilized a SQL injection attack against the rockyou.com website. RockYou provides applications for numerous social networking sites such as Facebook, MySpace, and Friendster, and thus included the associated login details created by users for those sites. The list contains over 32 million passwords. Due to the list's enormous size, we select typical sub-sets labeled as "the most common XX passwords", which "XX" stands for the number of passwords in the set. To make the experiment consistent with the PUF array structure we implemented, we divide the password string into groups with 4 characters (32 bits) of each group. And for each character, we map it first to ASCII code and then to binary value. For the input entropy, unlike the former evaluated bit entropy, we use the character entropy (8 bits) in this test. We calculate the entropy for each character and then average the result. Thus, in this case, the full entropy of a character should be 8 bits.

Table I: The Entropy results for different password sets.

Password Set	Input Entropy	Output Entropy	Input XOR Output
500-common	4.41	5.79	7.07
10k-common	4.44	6.14	7.23
100k-common	4.75	6.81	7.43
500-recommended	5.01	6.86	7.79

Table I shows the test results on the most common password sets from the RockYou list. Not surprising, the original entropy of is as low as 4 bits, which is around half of the full entropy. Several trends are evident from this plot. First, we observe the trend of initial input entropy increasing with increasing number of password set. Second, for each password set, there is an evident increase of entropy after processing by the PUF-passSE. Then, we exclusive-or the input with the output, the entropy increased above 7 bits which is near the full entropy as 8 bits. This notable results demonstrate that the performance of our PUF-passSE is good even if the input entropy is getting worse. Moreover, we test the 500 recommended passwords from the RockYou list. These kinds of 'good' passwords are selected by the password rules. For example, it should contain numbers intersecting with letters. The input entropy of this case is around 5 bits which is the highest for all the test cases. Although it is labeled by recommended passwords, it still shows a gap to the full entropy under our test. When applying the PUF enhancer on it, we can see a significant increase (from 5 bits toward 7.8 bits) in the entropy.

6.3. Mutual information of two devices

To prove that our proposed PUF holds uniqueness implemented in different devices, we test the mutual information (MI) of the outputs from two devices. In information theory, the mutual information (MI) is a measure of the mutual dependence between two variables. In the PUF-passSE application, we would like to find out the mutual dependence of two devices in order to show the generated password is unique to the device. If the mutual information of two devices is small, the password would seldom be same when two customers occasionally use the same initial password. That is to say, each device would use its unique hardware features in generating the passwords. Otherwise, the PUF-passSE would result in a collision when the input passwords are the same. Moreover, the attacker would not learn the passwords from another PUF-passSE device, if he can totally control one device.

The equation to calculate the mutual information of two devices is defined as:

$$\mathrm{I}(X,Y) = \mathrm{H}(X) - \mathrm{H}(X \mid Y) \qquad (4)$$

X represents the output of the Device 1 and *Y* represents the output of Device 2. The mutual information could be calculated as the entropy of one variable subtracting the conditional entropy on the other variable.

The results of the mutual information of two devices are shown in Table II. The first row is the input probability (p) selected as different test cases. From the second row, we find out that the mutual information of the two devices is less than 0.1 which is low compared to the entropy as around 0.9. Those results could demonstrate that by knowing the output of one Device, the attacker cannot get more information about the output of the other Device.

The limited mutual information among devices demonstrates the uniqueness of the PUF-passSE which is the significant advantage over the traditional cryptography algorithms, i.e, hash functions. As those functions are identical for different devices, the scheme implemented by hash functions would need to add extra random nonce to guarantee its uniqueness. However, in our PUF-passSE, the random function is embedded in the hardware structure which could save more hardware cost for the constrained IoT devices.

Table II: The mutual information for two devices

p	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
I(X,Y)	0.086	0.077	0.068	0.072	0.073	0.074	0.075	0.077	0.114

7. Conclusion

In this paper, we present a PUF-based entropy enhancer for the application of passwords. The design uses configurable ring oscillators (RO) PUF to generate randomness. We demonstrate that this approach can significantly increase the entropy of the source input. Besides, we also test the performance on real passwords set and show the achievements in pumping the entropy of the human generated passwords. Since the PUF-passSE is low-cost hardware add-ons to enhance the security level of passwords, incorporating such hardware device will provide a better and secure alternative to the IoT authentication applications.

Acknowledgement

This project is supported in part by AFOSR MURI under award number FA9550-14-1-0351. We also would like to express thanks for Dr.Xueyan Wang for her support on the Fig.2.

8. References

[1] Majzoobi, Mehrdad, Masoud Rostami, Farinaz Koushanfar, Dan S. Wallach, and Srinivas Devadas. "Slender PUF protocol: A lightweight, robust, and secure authentication by substring matching." In 2012 IEEE Symposium on Security and Privacy Workshops, pp. 33-44. IEEE, 2012.

[2] Wang, Qian, Mingze Gao, and Gang Qu. "A Machine Learning Attack Resistant Dual-mode PUF." Proceedings of the 2018 on Great Lakes Symposium on VLSI. ACM, 2018.

[3] Suh, G. Edward, and Srinivas Devadas. "Physical unclonable functions for device authentication and secret key generation." Proceedings of the 44th annual design automation conference. ACM, 2007.

[4] Johnson, Anju P., Rajat Subhra Chakraborty, and Debdeep Mukhopadhyay. "A PUF-enabled secure architecture for FPGA-based IoT applications." IEEE Transactions on Multi-Scale Computing Systems 1.2 (2015): 110-122.

[5] Zhang, Jiliang, Binhang Qi, Zheng Qin, and Gang Qu. "HCIC: Hardware-assisted Control-flow Integrity Checking." IEEE Internet of Things Journal (2018).

[6] Narayanan, Arvind, and Vitaly Shmatikov. "Fast dictionary attacks on passwords using time-space tradeoff." Proceedings of the 12th ACM conference on Computer and communications security. ACM, 2005.

[7] Ding, Jintai, Bo-Yin Yang, Chia-Hsin Owen Chen, Ming-Shing Chen, and Chen-Mou Cheng. "New differential-algebraic attacks and reparametrization of rainbow." In International Conference on Applied Cryptography and Network Security, pp. 242-257. Springer, Berlin, Heidelberg, 2008.

[8] Kumar, Himanshu, Sudhanshu Kumar, Remya Joseph, Dhananjay Kumar, Sunil Kumar Shrinarayan Singh, and Praveen Kumar. "Rainbow table to crack password using MD5 hashing algorithm." In Information & Communication Technologies (ICT), 2013 IEEE Conference on, pp. 433-439. IEEE, 2013.

[9] Shannon, Claude Elwood. "A mathematical theory of communication." Bell system technical journal 27, no. 3 (1948): 379-423.

[10] Florencio, Dinei, and Cormac Herley. "A large-scale study of web password habits." In Proceedings of the 16th international conference on World Wide Web, pp. 657-666. ACM, 2007.

[11] Brown, Alan S., Elisabeth Bracken, Sandy Zoccoli, and King Douglas. "Generating and remembering passwords." Applied Cognitive Psychology: The Official Journal of the Society for Applied Research in Memory and Cognition 18, no. 6 (2004): 641-651.

[12] Gao, Mingze, Khai Lai, and Gang Qu. "A highly flexible ring oscillator PUF." In Proceedings of the 51st Annual Design Automation Conference, pp. 1-6. ACM, 2014.

[13] Wang, Qian, and Gang Qu. "A Silicon PUF based Entropy Pump." IEEE Transactions on Dependable and Secure Computing (2018).

[14] Katz, Jonathan, and Yehuda Lindell. Introduction to modern cryptography. CRC press, 2014.

[15] Zhang, Yinqian, Fabian Monrose, and Michael K. Reiter. "The security of modern password expiration: An algorithmic framework and empirical analysis." In Proceedings of the 17th ACM conference on Computer and communications security, pp. 176-186. ACM, 2010.

[16] Farash, Mohammad Sabzinejad, and Mahmoud Ahmadian Attari. "An efficient and provably secure three-party password-based authenticated key exchange protocol based on Chebyshev chaotic maps." Nonlinear Dynamics 77, no. 1-2 (2014): 399-411.

[17] Ding, Jintai, Saed Alsayigh, Jean Lancrenon, R. V. Saraswathy, and Michael Snook. "Provably secure password authenticated key exchange based on RLWE for the post-quantum world." In Cryptographers' Track at the RSA Conference, pp. 183-204. Springer, Cham, 2017.

[18] Almeshekah, Mohammed H., Christopher N. Gutierrez, Mikhail J. Atallah, and Eugene H. Spafford. "Ersatzpasswords: Ending password cracking and detecting password leakage." In Proceedings of the 31st Annual Computer Security Applications Conference, pp. 311-320. ACM, 2015.