Novel PUF-Based Authentication Protocol for IoT Devices with Secure Boot and Fuzzy Matching

Hoang-Long Pham[†], Thuy-Quynh Tran-Thi[‡], Duy-Hieu Bui[†], and Xuan-Tu Tran[†]
[†]*VNU Information Technology Institute*
[‡]*VNU University of Engineering and Technology*
144 Xuan Thuy Road, Cau Giay District, Ha Noi, VietNam
Corresponding author's email: hieubd@vnu.edu.vn

***Abstract*—Traditional security solutions such as public key cryptosystems can provide a high level of security but are costly regarding memory footprint, computational complexity and power consumption. Therefore, they are unsuitable for Internet of Things devices with limited resources. One way to improve the security of IoT systems is to use a Physical Unclonable Function (PUF), a unique digital fingerprint of each device. This paper proposes a lightweight authentication protocol based on Strong PUFs with fuzzy matching and secure boot for IoT devices. The proposed system containing a RISC-V subsystem, a block cipher module and an Arbiter PUF has been successfully implemented on Xilinx FPGA Development Kit Arty A7 100T.**

***Index Terms*—Physical Unclonable Function, Internet of Things, Authentication protocol**

I. Introduction

With the rapid growth of the Internet of Things (IoT) and the proliferation of interconnected devices, ensuring the security of devices has become a top concern. However, traditional security solutions are not suitable for IoT applications due to limited computing resources, low memory footprint and ultra-low power consumption. In addition, production cost is also an important factor. Therefore, lightweight security protocols are crucial to improve system security and reduce production costs for constrained IoT devices.

To address this challenge, one approach is to leverage Physical Unclonable Function (PUF) to enhance security for IoT systems. Introduced in 2002 by Gassend *et al.* in [1], PUFs are a hardware-based security primitive that utilizes the unique manufacturing variations in each device to generate an unclonable hardware fingerprint. This makes PUFs superior to other hardware-based security schemes since hackers cannot clone the intrinsic device properties. PUFs enable device-based identification and authentication and can provide a low-cost alternative for the on-demand generation of cryptographic keys in IoT devices. This contrasts the conventional methods where the server produces and distributes secret keys which are also stored in IoT device non-volatile memories. Therefore, PUFs offer a unique and effective solution to the security challenges many IoT devices face. By leveraging PUFs, IoT devices can achieve strong security assurance with low power consumption and area cost.

This paper proposes a PUF-based authentication protocol based on LoRaWAN 1.0.x that allows fuzzy matching and secure boot. LoRaWAN, a widely adopted Low-Power Wide-Area Network (LPWAN) technology specifically designed for IoT applications. LoRaWAN provides a reliable and energy-efficient communication platform, enabling long-range connectivity and cost-effective data transmission for IoT devices. Additionally, the protocol incorporates Advanced Encryption Standard (AES) cryptographic algorithms to further secure data exchanged between devices and the network server. AES ensures the integrity and confidentiality of data transmitted over the LoRaWAN network, enhancing the overall security of the proposed authentication protocol.

Unlike other protocols that closely compare CRP pairs obtained during authentication with CRPs stored in the server's database, our protocol allows for small differences between CRP pairs. This flexibility allows for strong authentication even in situations where the CRP may not match completely due to environmental factors or the aging process of the device. In addition, IoT devices are often placed in sensitive, less-protected places. If an attacker steals that device, they can use challenges and generate responses which can be used to break the system security. The secure boot solves the above-mentioned problem by only allowing "trusted software" to read/write the PUFs. The main advantage of our proposed protocol lies in its ability to maintain security requirements without using a "high-performance" PUF architecture. By eliminating the need for complex and resource-intensive PUF implementations, we significantly reduce the cost and energy consumption associated with PUFs used in the authentication process.

The rests of this paper are organized as follows. Section II presents the related studies to our work. Section III discusses the fundamentals of PUFs and our proposed protocol. It also explains the concept of fuzzy matching and provides a detailed look at our security protocol. The security analysis and performance analysis will be presented in Section IV. Section V introduces a RISC-V system including an Arbiter PUF and an AES encryption accelerator to implement the proposed protocol on Arty-A7-100T FPGA Development Board. Finally, there are some conclusions and perspectives in Section VI.

II. Related Works

In this section, we will discuss some related works that address lightweight authentication challenges in IoT systems, focusing on the use of Physical Unclonable Functions (PUFs). Various security challenges for IoT systems are outlined in [2]. The author recommends using PUFs to secure IoT systems. In [3] and [4], the authors proposed a mutual authentication protocol. The work in [3] uses an ID tag which is encrypted with a hash function and a stream-cipher-based OTP by Challenge and Response Pair (CRP) of a PUF. The proposed protocol uses the PUF output to generate a transient key dynamically. The authors in [4] proposed a protocol that uses the CRP of the node's PUF to verify the node with the sink. The two above protocols require the stable CRP received from the device's PUF. However, CRPs of a PUF might have a marginal variation under different working conditions.

In [5], the author provided a PUF-based mutual authentication method between devices and devices to enhance the security of IoT. This work used only one initial CRP stored in the authentication server's database. CRP for future authentication is updated during the execution of the authentication process. This arrangement minimizes server storage space and solves management problems. Its purpose is to avoid the CRPs being exposed when the server is attacked.

In [6], the authors proposed a three-phase authentication protocol. A new encoding and decoding method enables strong PUF for key generation and exchange using machine learning. Their protocol can recover an original key using no public helper data. They proposed to use a strong PUF with increased complexity, such as XOR Arbiter PUF. To provide the capability of generating a large number of encryption keys, they proposed using a machine learning-based equivalent model of the PUF on a Trusted Third Party (TTP) verifying server. The machine learning model of the PUF, in turn, gives access to its full CRP space with some negligible misprediction probability. The use of a Trusted Third Party protects PUF from being accessed by untrusted software.

In [7], almost the same as in [5], the authors proposed a private PUF-based Anonymous Authentication scheme that overcomes the shortcomings of the prior attempts to incorporate PUFs for anonymous authentication schemes. Traditional PUF-based authentication protocols have their limitations as they only work based on challenge-response pairs (CRPs) exposed to the verifier, thus violating the principle of anonymity. There, they ensured that even if the PUF instance is private to the user, it can be used to authenticate the application provider. Besides, no raw CRPs need to be stored in a secure database, thus making it more difficult for an adversary to launch modelling attacks on the deployed PUFs. In this protocol, authors have proposed a private PUF-based scheme that overcomes the shortcomings of the Trusted Platform Module to incorporate PUF for Anonymous Authentication schemes.

In [8], the authors propose a lightweight PUF-based authentication protocol. That was implemented on a wireless sensor network constructed using the resource-limited IoT devices: Zolertia Zoul re-mote. The functionality of the proposed scheme was verified using a server-client configuration. The authors have used the PUF model instead of the PUF module, the protocol requires the accuracy CRPs at the authentication time and the CRPs stored in the server.

TABLE I
NOTATIONS USED IN OUR PROTOCOL

Notation	**Significance**
$\oplus$	XOR operator
C_0, C_1	Challenge for authentication
R_0, R_1	Response of PUF according to C_0, C_1

In conclusion, the existing researches offer valuable insights into the IoT device authentication protocols, highlighting the distinct advantages presented in each approach. Drawing upon the findings from the above analysis, we now introduce a novel authentication protocol that leverages the strengths gleaned from the reviewed articles. Through careful synthesis and integration, our proposed protocol aims to create a more robust and efficient authentication mechanism for IoT devices, addressing the evolving challenges in this domain.

III. Proposed protocol

This paper presents our approach for IoT device authentication based on PUFs. The proposed protocol provides a more efficient authentication scheme to prevent unauthorized access to the PUFs if the device is accidentally stolen. This can be done using secure boot loader that only allows the authenticated firmware to run on the device. After the device is securely booted, it will be able to join the network using the inputs and outputs from the PUF.

In this work, the PUF is integrated into the System-on-Chip and the software is protected by the secure boot in the boot firmware. Only the authenticated firmware can be run in the system. Therefore, the attackers can not manipulate the inputs of the PUF and read the output. The server is assumed to be trusted. Generally, the servers are not resource-constrained devices. Therefore, they can be protected by the current state-of-the-art security countermeasure available.

In addition, fuzzy matching is applied in the authentication process so that PUF can be used even under extreme conditions where the PUF outputs might have a margin variation. PUF usage time is prolonged even when it is affected by device aging. The proposed method does not require "high-performance" PUF architectures resulting in cost savings.

A. The proposed protocol

This section describes our proposed authentication protocol. Table I lists the notations that appear in our protocol. Those notations are explained.

As described above, the proposed protocol is developed based on LoRaWAN 1.0.x in combination with the secure boot to protect the PUF. Accordingly, the device's PUF can avoid being manipulated by a malicious program to extract the response if an attacker accidentally steals it. Fig. 1 below describes the steps in the authentication protocol in details.

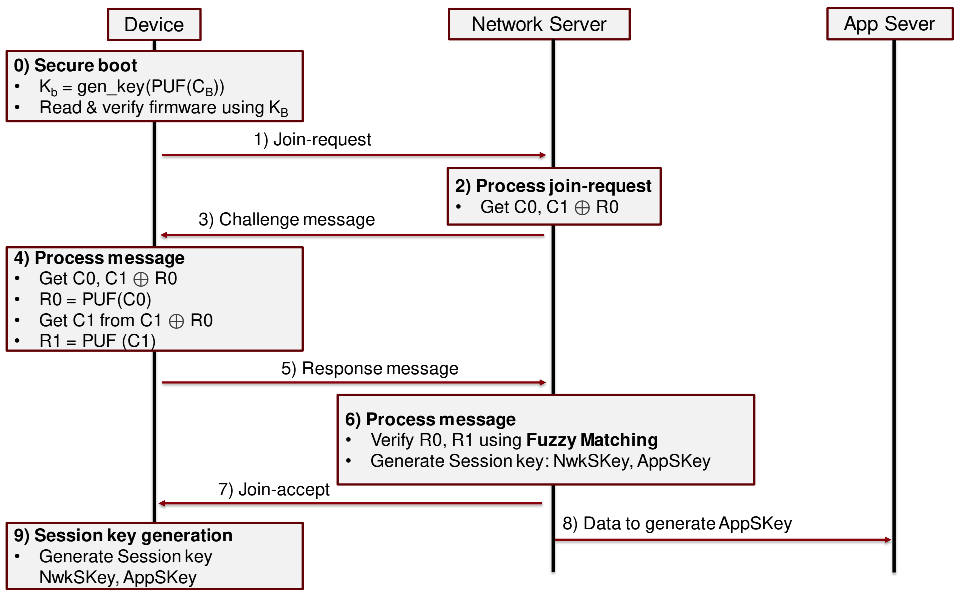


Fig. 1. The proposed authentication protocol

1) Secure boot: When the device starts up, the boot code preinstalled in the silicon starts the system by reading the challenge C_B from the flash memory and using it to generate the boot key K_B. Then, the boot code will use K_B to decrypt the firmware and verify its integrity on the flash memory. If the firmware is authenticated, the boot code runs the firmware. After that, the system will start the authentication process to join the network.

2) Join Request: In the first step in the authentication process, the device sends a join request message to the network server. The message consists of *AppEUI* (8 bytes), *DevEUI* (8 bytes) and *DevNonce* (2 bytes). The join request is not encryption but authenticated using a preinstalled key *AppKey*.

3) Process Join Request: The network server verifies the integrity of the join request using *AppKey*. The network server can use the device ID *DevEUI* to access the CRP database, if the join request is valid. The network server selects two CRPs (C_0, R_0) and (C_1, R_1). It then constructs $A_0 = C_1 \oplus R_0$.

4) Send Challenge Message: The network server sends the challenge messages encrypted by *AppKey* back to the device including C_0 and A_0.

5) Process Challenge Message: When the device receives the challenge message, it decrypts the challenge message by the *AppKey* and gets C_0 and A_0. The device uses the PUF to generate R_0 from C_0. Then it calculates $C_1 = A_0 \oplus R0$. The next step uses the PUF to generate R_1 from C_1.

6) Send Response Message: The device encrypts the response message consisting of R_0 (8 bytes), R_1 (8 bytes), along with *DevNonce* (2 bytes) by using *AppKey*. It sends the encrypted response messages back to the Network Server.

7) Process Response Message: The network server decrypts the response message by *AppKey* to get R_0, R_1 and verifies them using Fuzzy Matching techniques. If that response is unsatisfactory, the network server rejects that device. Else, the device is accepted. The network server deletes two challenge response pair CRP_0, CRP_1. Then the network server generates two session keys (*NwkSKey* and *AppSKey*) using the information in the join accept message. These two keys are used to encrypt the network traffic and the application data.

8) Send Join accept: The device also needs to generate the session keys. Therefore, the network server sends the join accept message. Join accept message consists of *AppNonce* (3 bytes), *NetID* (3 bytes), *DevAddr* (4 bytes), *DLSettings* (1 byte), *RXDelay* (1 byte), *CFList* (optional-16 bytes). The join accept message itself is then encrypted with the *AppKey*.

9) Key distribution: The network server keeps the *NwkSKey* and distributes the *AppSKey* to the application server.

10) Device generates session key: The end device decrypts the join accept message. The end device uses the information in the join accept message to derive the two session keys, the Network Session Key (*NwkSKey*) and the Application Session Key (*AppSKey*). The device is now activated on the network and can start transmitting data.

IV. Security Analysis

The proposed protocol is based on PUFs, and the security analysis is presented below.

1) Secure boot: In this protocol, we used a secure boot mechanism to prevent malicious third parties from accessing device PUF. Only the device processor can access the PUF using the authenticated firmware. Consequently, the secure boot mechanism does not allow a third party to modify the firmware to create the CRP database when stealing the device.

2) Mutual authentication: This protocol provides mutual authentication because the device can verify that the network server knows R_0. On the other hand, the network server can verify that the device knows R_1. The fuzzy matching technique

TABLE II
COMPARISON WITH OTHER PROTOCOLS

Protocol features	[3]	[4]	[5]	[6]	[7]	[8]	**This work**
Mutual Authentication	v	v	v	v	v	v	v
Integrity	×	×	×	v	v	v	v
Confidentiality	v	×	×	v	v	v	v
Replay attacks	v	v	v	v	v	v	v
Eavesdropping	v	v	v	v	v	o	v
De-synchronization	v	?	?	v	?	?	v
Server Spoofing	v	v	v	v	v	v	v
Client Impersonation	v	v	v	v	v	×	v
Man in the middle	?	?	?	?	?	?	v
Modification	v	v	×	v	v	×	v

v: Satisfied ×: Not Satisfied o: Partly Satisfied ?: Not evaluated criteria

allows a small variation due to different working environments that might affect the PUF's outputs.

3) Integrity and confidentiality: The proposed protocols use AES-CCM for authenticated encryption with associated data (AEAD). Except for the join request message, which is not encrypted but authenticated, all the other messages between the device and the server use authenticated encryption with associated data (AEAD) to implement integrity and confidentiality.

4) Replay attacks, eavesdropping, and desynchronization attacks: The protocol includes the usage of nonces, which helps prevent replay attacks. Each transaction involves the exchange of a nonce between the device and the server, ensuring that an attacker cannot replay the messages. This countermeasure helps maintain the freshness and integrity of the communication. The device and network server track the nonce so that the key will need to be changed when the nonce repeats. In addition, the data sent by the device in the network are encrypted and authenticated. Therefore, the attackers can eavesdrop on the packet but can not decrypt it without knowing the session keys. Desynchronization attacks are also impossible because all the system's values are transmitted using authenticated encryption with authenticated data.

5) Server spoofing, client Impersonation and man-in-the-middle attacks: The proposed protocol also resists server spoofing, client impersonation and man-in-the-middle attacks because it can authenticate both the device and the server. The device can ensure that the server knows the *AppKey* in advance, and has the CRP database. If the attacker wants to perform server spoofing attacks, they must have these two values. The CRP database on the server can be further encrypted on the server to enhance security.

On the other hand, if the attackers want to use a fake device, they must know *AppKey*, R_0 and R_1. However, because C_0 and C_1 are selected at random, they must know a large CRP database. This is hard because the secure boot protects the AppKey and the PUF.

Finally, man-in-the-middle attacks are protected using authenticated encryption with associated data using AES-CCM. The *AppKey* is used to derive the session keys. It is hard for attackers to fake these two keys because the protocol generates it through an encrypted communication channel.

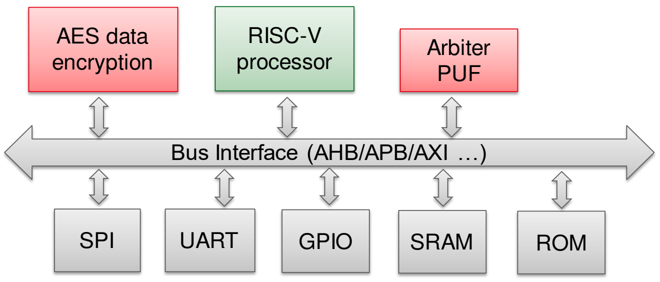


Fig. 2. System architecture.

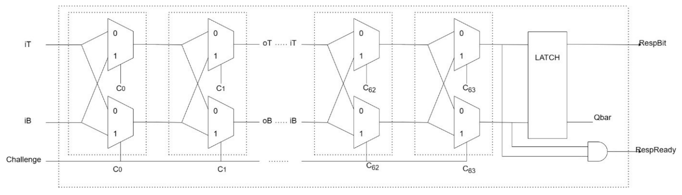


Fig. 3. Delay cells for the Arbiter PUF.

6) Fuzzy matching: The use of fuzzy matching in the PUF-based authentication protocol offers certain cost and system performance benefits. Fuzzy matching allows the protocol to handle variations and slight differences in PUF responses, making the authentication process more tolerant to environmental factors and device aging. This enhances the overall performance and reliability of the system by accommodating variations in PUF outputs, even under extreme PVT conditions. Consequently, the system can maintain its effectiveness over an extended period of time without requiring frequent recalibration or replacement.

7) Comparisons: Table IV presents the comparison of our proposed protocol with the others' works. It is clear that our proposed protocol can resist most of the attacks on the authentication protocol using PUFs for IoT. Some protocols use Hash Function to ensure data integrity and Asymmetric Encryption for authentication. However, these cryptographic mechanisms are not lightweight and not suitable for constrained devices. Our protocol, in contrast, uses only the lightweight symmetric primitive to enhance IoT device security.

V. Hardware Implementation

Our IoT system, which implements the proposed protocol, uses a 64-bit Arbiter PUF for authentication and an AES core for key generation and encryption. The PUF and AES modules are packaged as IPs, integrated into a RISC-V SoC, and then implemented on FPGA Arty A7 100T Development Board. Fig. 2 shows our proposed system architecture with the PUF and AES module for data encryption. The AES module is reused from the work in [9].

The PUF architecture used in this work is Arbiter PUF, a strong PUF. The PUF contains 64 switch chains which can generate 64-bit responses for a 64-bit challenge. A switch chain can be constructed from multiplexers and a latch, as described in Fig. 3. The architecture of the Arbiter PUF is

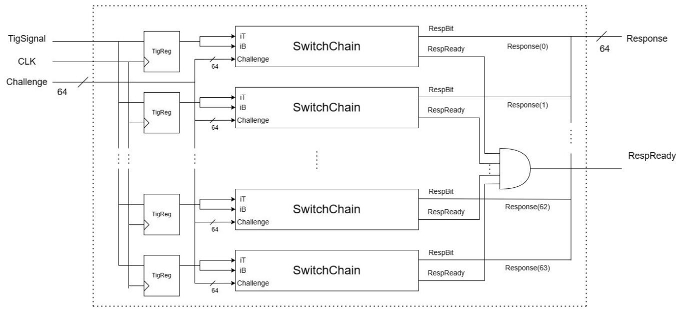


Fig. 4. 64 switch chains used in the Arbiter PUF.

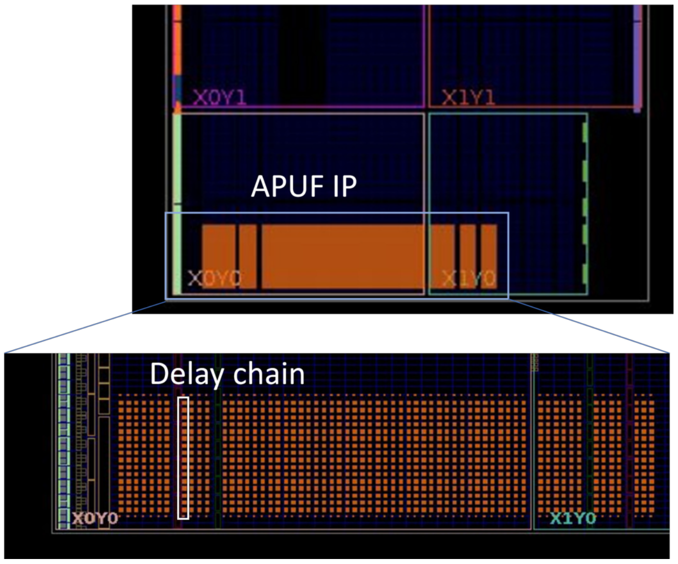


Fig. 5. Arbiter PUF when implemented on FPGA.

shown in Fig. 4. It contains 64 switch chains to construct 64-bit outputs. Each chain might have different delays. Therefore, the 64-bit output will be latched only when all switch chains have finished the delay propagation process.

To enhance the quality of the Arbiter PUF on FPGA, the delay cells need to be mapped into a specific pattern on FPGA. In this work, the multiplexer in Fig. 3 is mapped directly into the FPGA LUT primitive. Each cell is placed in a specific FPGA location to form the targeted pattern. The implementation of the Arbiter PUF on FPGA is shown in Fig. 5. Each delay chain is organized vertically. It is clear that the Arbiter PUF was implemented as expected.

The whole system in Fig. 2 has been implemented on Arty A7 100T FPGA development kit using Xilinx Vivado. Fig. 6 shows the breakdown of the hardware utilization of each module on FPGA. It is obvious that the Arbiter PUF use a few registers but a large number of LUTs as multiplexers. Overall, it occupies about 14$\%$ of the total number of slices. In addition, the AES encryption core occupied about 13$\%$ of the slices. The biggest part of the system is the RISC-V core with 37$\%$ of the overall system resource. The second biggest part is the peripherals with 30$\%$.

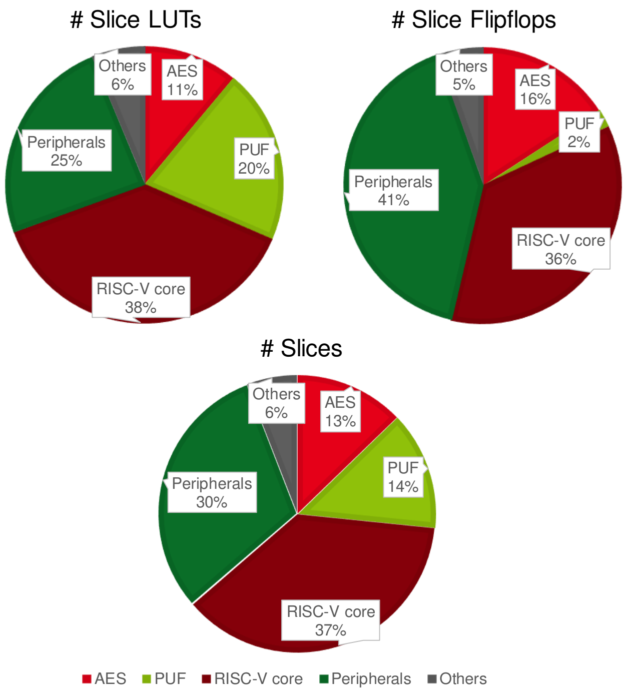
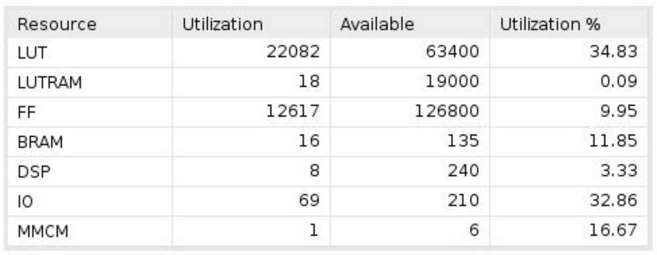


Fig. 6. Hardware utilization of each module on FPGA.



Resource	Utilization	Available	Utilization %
LUT	22082	63400	34.83
LUTRAM	18	19000	0.09
FF	12617	126800	9.95
BRAM	16	135	11.85
DSP	8	240	3.33
IO	69	210	32.86
MMCM	1	6	16.67

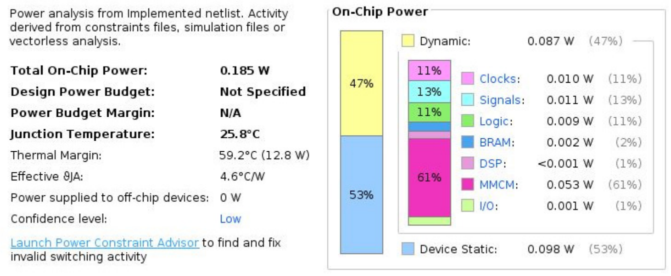
Fig. 7. System hardware utilization on Arty A7 100T development Kit.

Fig. 7 describes the entire system's occupied resource on Arty A7 100T FPGA development kit. It is obvious that the system occupies a small portion of the development kit. It has room to integrate more designs into this system. Overall, the design takes 35$\%$ of LUTs, and about 10$\%$ of the flipflops on the boards. In addition, the system uses 16 BRAMs, 8 DSPs and 69 I/Os. It is clear that the proposed system is lightweight with a small hardware implementation area.

Last but not least, the power consumption on FPGA is also reported using Xilinx Vivado design tools. The power consumption of the proposed design is presented in Fig. 8. The overall system consumes about $185mW$. Dynamic power consumption is about $87mW$, while static power consumption is $98mW$. It is clear that the proposed design has low power consumption. The processor runs at $25MHz$.

VI. Conclusions

This paper proposed a lightweight authentication protocol with lower power consumption and resource usage, enhancing


Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	**0.185 W**
Design Power Budget:	**Not Specified**
Power Budget Margin:	**N/A**
Junction Temperature:	**25.8°C**
Thermal Margin:	59.2°C (12.8 W)
Effective ϑJA:	4.6°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

🟨	Dynamic:	0.087 W	(47%)
🟪	Clocks:	0.010 W	(11%)
🟦	Signals:	0.011 W	(13%)
🟩	Logic:	0.009 W	(11%)
🟦	BRAM:	0.002 W	(2%)
🟪	DSP:	<0.001 W	(1%)
🟪	MMCM:	0.053 W	(61%)
🟩	I/O:	0.001 W	(1%)
🟦	Device Static:	0.098 W	(53%)

Fig. 8. Power consumption on FPGA

security with fuzzy matching and secure boot. The basic idea is to use the AES core and PUF's output to generate the key for secure boot. The secure boot will enhance the security of PUF by preventing untrusted third parties from accessing PUF to extract CRPs. Fuzzy matching will help save costs for PUF design because it does not require too complex, high-performance PUF architectures. Our proposed design was successfully implemented on Arty A7 100T development kit. The hardware implementation results show that the proposed system had a small hardware area footprint with a low power consumption of $185mW$. The proposed solution can be used for constrained IoT systems.

Acknowledgment

This research has been partly supported by Vietnam National University, Hanoi under the project No. QG.22.70.

References

[1] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Silicon physical random functions," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, CCS '02, (New York, NY, USA), p. 148–160, Association for Computing Machinery, 2002.

[2] T. Xu, J. B. Wendt, and M. Potkonjak, "Security of iot systems: Design challenges and opportunities," in *2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 417–423, 2014.

[3] B. Kim, S. Yoon, Y. Kang, and D. Choi, "Puf based iot device authentication scheme," in *2019 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 1460–1462, 2019.

[4] Y. S. Lee, H. J. Lee, and E. Alasaarela, "Mutual authentication in wireless body sensor networks (wbsn) based on physical unclonable function (puf)," in *2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pp. 1314–1318, 2013.

[5] S. Yoon, B. Kim, Y. Kang, and D. Choi, "Puf-based authentication scheme for iot devices," in *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 1792–1794, 2020.

[6] A. Ali-pour, F. Afghah, D. Hely, V. Beroulle, and G. Di Natale, "Secure puf-based authentication and key exchange protocol using machine learning," in *2022 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 386–389, 2022.

[7] U. Chaterjee, D. Mukhopadhyay, and R. S. Chakraborty, "3paa: A private puf protocol for anonymous authentication," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 756–769, 2021.

[8] Y. Yilmaz, S. R. Gunn, and B. Halak, "Lightweight puf-based authentication protocol for iot devices," in *2018 IEEE 3rd International Verification and Security Workshop (IVSW)*, pp. 38–43, 2018.

[9] N.-D. Nguyen, D.-H. Bui, and X.-T. Tran, "A lightweight aead encryption core to secure iot applications," in *2020 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, pp. 35–38, 2020.