IBIMA Publishing
Communications of International Proceedings
https://ibimapublishing.com/p-articles/46SP/2025/4641525/
Vol. 2025 (37), Article ID 4641525

Combining Lightweight Cryptographic Protocols with Physically Unclonable Functions (PUFs) for Key Generation and Device Authentication in the Industrial Internet of Things (IIoT)*

Anna GROCHOLEWSKA-CZURYLO and Agata DABROWSKA

Poznan University of Technology, Poznan, Poland

Correspondence should be addressed to: Anna GROCHOLEWSKA-CZURYLO, anna.grocholewska-czurylo@put.poznan.pl

* Presented at the 46th IBIMA International Conference, 26-27 November 2025, Ronda, Spain

Abstract

With the increased scale of Industry 4.0 systems, security of Industry Internet of Things (IIoT) became a critical challenge. Many devices still use legacy cryptographic algorithms which are often too "heavy" for limited resources of most hardware platforms. In this work, a hybrid approach is investigated - a combination of lightweight cryptographic protocols with Physically Unclonable Functions (PUF) used for key generation and device authentication.

The aim was to verify if such integration may bring practical security benefits without putting extensive load on the embedded systems. Four lightweight protocols have been tested - Speck, Ascon-128, EAP and Kerberos - both by themselves, as well as in combination with PUF. Each implementation has been evaluated for processing time, memory footprint and resilience to attacks and errors.

Results show a clear compromise - adding PUF improves key uniqueness and resilience to modelling attacks. However, it slightly lowers error tolerance. Kerberos turned out to be the most stable, while Speck was the fastest. In general, hybrid model implementations seem to be feasible and promising for real-world IIoT deployments. Further verification of hardware implementation is required to confirm usability in noisy industrial environments.

Keywords: Industrial Internet of Things, Lightweight Cryptography Protocols, Reduced Capabilities devices, PUF

Introduction

In today's fast-changing industrial landscape, securing assets, production systems, and communication networks has become growing concern [1]. The Industrial Internet of Things (IIoT), now widely used across factories and automation environments, is especially vulnerable to various types of cyber threats [2], [3], [4]. Protecting critical operations and sensitive data has therefore become not just a priority, but a necessity.

IIoT devices typically operate under strict resource limitations - low processing power, small memory capacity and constrained energy budgets. Any security mechanisms added to such systems must therefore remain lightweight and efficient. Still, these restrictions cannot come at the expense of protection, in fact, given the sensitivity of industrial data, device and network security must often be stronger, than in traditional IT systems.

Cite this Article as: Anna GROCHOLEWSKA-CZURYLO and Agata DABROWSKA, Vol. 2025 (37) "Combining Lightweight Cryptographic Protocols with Physically Unclonable Functions (PUFs) for Key Generation and Device Authentication in the Industrial Internet of Things (IIoT)" Communications of International Proceedings, Vol. 2025 (37), Article ID 4641525, https://doi.org/10.5171/2025.4641525

Several approaches have been proposed to address these challenges, including **Physically Unclonable Functions (PUFs)** and **Lightweight Cryptographic Protocols**, both designed for constrained environments. Yet, despite the progress made, combining these two technologies into single, coherent security architecture for IIoT is still uncommon.

The purpose of this study is to examine the feasibility of integrating PUFs with lightweight cryptographic methods, and to evaluate how such hybrid approach compares to existing security mechanisms currently applied in the industrial domain.

Literature Review

In the era of Industry 4.0, the demand for robust security has become increasingly evident. The integration of smart devices, sensors, and embedded systems has introduced numerous security challenges, that must be addressed through advanced security mechanisms. As industrial systems become more interconnected, they are increasingly vulnerable to cyber threats, counterfeiting, and unauthorized attacks. Traditional cryptographic methods, while effective, often require substantial computational power and storage, making them difficult to implement in resource-constrained Industrial Internet of Things (IIoT) devices.

Physically Unclonable Functions (PUFs) have emerged as promising hardware-based solution for authorization and authentication in IIoT networks. They are considered lightweight, economical, and ubiquitous, enabling secure authentication without adding complex systems or extra resources. PUFs are generally classified into two categories - weak and strong - depending on parameters, such as uniqueness and the number of Challenge-Response Pairs (CRPs). Despite their advantages, PUFs remain vulnerable to certain attacks such as Man-in-the-Middle (MitM).

Several studies propose enhanced PUF implementations to address these weaknesses. The Virtual PUF (VPUF) proposed in [5] uses split-learning approach with an encoder-decoder architecture to mitigate vulnerabilities of physical PUFs. This method significantly reduces computational requirements and energy consumption while maintaining 100 % secure authentication. However, it has not yet been validated in industrial environments.

Quantum-based PUFs have also been explored as a potential advancement. Bathalapalli [6] presents a Security-by-Design solution utilizing Quantum Logic Gates on IBM Quantum hardware. Quantum PUFs can generate large numbers of CRPs and provide robust device identities, but their implementation may be too demanding for resource-limited IIoT devices.

In the context of Post-Quantum Cryptography (PQC), [7] introduces a Frequency Adjustable Software PUF (FAS-PUF) to enhance PQC chip security. Operating within the timing logic of R-LWE decryption circuits, FAS-PUF maintains strong uniqueness, uniformity and stability, offering a viable software alternative to traditional hardware implementations.

The cryptographic perspective on IIoT security is discussed in [8], where Mishra emphasizes that classical cryptographic algorithms securing IoT infrastructure are threatened by advances in quantum computing. The study identifies a gap in research on PQC schemes tailored for resource-constrained environments and examines the suitability of existing microcontrollers for such implementations.

Hybrid PUF-PQC solutions are also being investigated. Cultice and Thapliyap [9] propose a PUF-based Post-Quantum Cryptographic framework for vehicular communication (CAN-FD), demonstrating improved resistance to cryptographic attacks. Although promising, this approach has not yet been assessed in industrial contexts.

A broader overview of IoT and IIoT vulnerabilities is presented in [10], which also discusses the potential of blockchain-based mechanisms such as Tangle to address security issues in centralized architectures. The study highlights both, advantages and limitations of such distributed approaches in industrial environments.

Further integration between PUFs and PQC is examined in [11], where CRYSTALS-Kyber - the NIST selected post-quantum encryption algorithm - is combined with SRAM PUFs for key generation on ARM architectures. The approach enhances entropy and physical security, but remains computationally heavy for IIoT devices.

Similarly, the CAKE-PUF protocol [12] combines PUF-based authentication with Elliptic Curve Diffie-Hellman (ECDH) key exchange, to deliver secure, resource-efficient communication suitable for smart factories. By integrating ECDH, the protocol mitigates MitM attacks while maintaining efficiency in constrained environments.

Overall, existing literature explores multiple security paradigms combining PUFs with Post-Quantum Cryptography, blockchain, and machine-learning-based enhancements. While each method addresses specific aspects of IIoT security, none provide comprehensive solution that jointly ensures postquantum resilience, lightweight performance, and hardware-level protection. This gap underscores the need for an integrated approach that leverages the advantages of both PUFs and efficient cryptographic mechanisms to secure resource-constrained industrial environments.

The rest of the paper is organized as follows: In Methods (Section 3), we describe the methodology of the study and simulation environment. Section 4, Experiments, describes the framework and experimental tests that were performed. In Results (Section 5), the obtained outcomes are presented. Lastly, Conclusions (Section 6) summarizes the findings and proposes directions for future research.

Methods

Research methodology focuses on designing, implementing, and testing a hybrid cryptographic architecture that combines Physically Unclonable Functions (PUFs) with lightweight cryptographic protocols, without changing the core logic of existing communication frameworks. The goal of the hybrid system is to enhance security, while maintaining compatibility with the strict resource constraints of embedded and edge devices.

PUFs enable the on-demand generation of cryptographic keys directly in hardware, eliminating the need for key storage in memory. This significantly increases resistance to physical tampering and key extraction attacks. At the same time, lightweight cryptographic protocols ensure encryption, authentication and data integrity, with minimal computational and energy overhead.

The main objective of this architecture is to provide a dynamic, non-stored cryptographic key, by integrating PUFs with lightweight protocols. As a result, system achieves stronger protection against both software-based and physical attacks, while remaining suitable for constrained IIoT environments.

Algorithm of a Lightweight Cryptographic Protocol with PUF Integration:

1. System startup.
2. Challenge generation.
3. Key creation using PUF by generating a response to the challenge.
4. Encryption or authentication using the lightweight cryptographic protocol with the PUF generated key.
5. Response verification.
6. Program termination.

The architecture of the hybrid system remains simple - it only requires replacing the conventional key used in the protocol with one generated by the PUF.

To implement the Physically Unclonable Function, the *pypuf* library was used [14][15]. The PUF applied in this work is based on the Arbiter PUF, classified as a strong PUF. According to prior research [13], Arbiter PUFs are particularly well-suited for Industrial IoT applications.

Four lightweight cryptographic protocols were selected: Speck and Ascon-128 for encryption, and EAP and Kerberos for authentication. Each protocol was implemented following its respective specification.

Four tests were performed: (1) **Processing Time Test** using *timeit*, (2) **Memory Usage Test** using *memory_profiler*, (3) **Attack Resilience Test** using logistic regression (ML model), and (4) **Error Resilience Test** measuring the Hamming distance between noisy and ideal PUF outputs. Each test was conducted for N trials (100 to 500, in steps of 100) to ensure statistical reliability.

All implementations were developed in Python 3.13.3. The simulations are software-based and their purpose is to evaluate functionality, performance, and resilience under controlled experimental conditions.

Experiments

The implemented hybrid solutions combining Lightweight Cryptographic Protocols with Physically Unclonable Functions (PUFs) were checked with the following four tests. Two of them evaluated performance - processing time and memory usage - and the remaining two assessed system resilience: attack resilience and error resilience. Each result represents the average value obtained from *N* program trials. The experiments were conducted for 100, 200, 300, 400, and 500 iterations to observe trends as the number of repetitions increased. All tests were performed for the Lightweight Cryptographic Protocols both **with** and **without** the PUF-generated key to ensure comparative evaluation.

For hybrid configurations - integrating the Lightweight Protocol with PUF-based key generation - the following PUF parameters were used:

- number of challenges: 1024
- length of challenge: 64
- seed: 1 + *i* (where *i* istrial number)
- noise for the fault tolerance test: 0 (as recommended scope of the *pypuf* library authors [14])

Processing Time Test

The Processing Time Test measures the average execution time of the encryption or authentication functions. The *timeit* library was used for precise timing. For each of *N* trials, the total execution time was recorded, and the mean processing time was calculated. The algorithmic structure follows standard benchmarking procedures, as illustrated in *Algorithm 1*.

Algorithm 1. Processing Time Test

1. Initialize an empty list *times*
2. For *i* to *N* repeat steps 3 - 6
3. Start the time measurement
4. Start the function (encryption or authentication)
5. End the time measurement
6. Add the differences between the end of the time measurement and the start of the time measurement to the *times* list
7. Calculate the average of the *N* trials from the *times* list
8. Display information about the result of the average execution time of the function from *N* trials

Memory Usage Test

The Memory Usage Test evaluates the amount of RAM required to execute each function across *N* trials. Measurements were performed using the *memory_usage* function from the *memory_profiler* module.

For each trial, the peak memory consumption was recorded, and the average memory usage was computed over all trials. A detailed overview of the test implementation is provided in *Algorithm 2*.

Algorithm 2. Memory Usage Test

1. Initialize an empty list of *memoryUsages*
2. For *i* to *N* repeat steps 3 - 4
3. Run memory_usage for the selected function
4. Save the maximum memory usage for *i* trial in the *memoryUsage* list
5. Calculate the average of *N* trials from the *memoryUsages* list
6. Display the result of the average memory usage needed to execute the function for *N* trials

Attack Resistance Test

Attack Resistance Test tests the resistance of the protocol to the attack based on Machine Learning. To perform the test, the LogisticRegression from *sklearn.linear_model* was used, which is responsible for performing logical regression, which is used to build a machine learning model for attack simulation. The *train_test_split* function from *sklearn.model_selection* is used to divide the dataset into training and testing sets. Finally, the accuracy_score function from sklearn.metrics is used to evaluate the success rate of the machine learning attack. The result of the test is the average of *N* samples, which indicates to what extent the function is resistant to the attack. The test procedure is shown in *Algorithm 3*.

Algorithm 3. Attack Resilience Test.

1. Initialize an empty *accuracies* list
2. For *i* to *N* repeat steps 3 - 7
3. Generate input data
4. Split data into training and test sets
5. Train ML model on training data
6. Check model accuracy for test data
7. Save accuracy from *i* sample to the *accuracies* list
8. Calculate the average of *N* trials from the *accuracies* list
9. Display information about the resistance to attacks result for the function from *N* trials

The Error Resilience

The Error Resilience Test checks the extent to which the responses differ for keys with different noise levels – one with an "ideal" noise level of 0.0, the other with a noise level of 0.2. The difference is calculated as Hamming Distance, and the test result is the average Hamming Distance between the function responses for the test with *N* trials. This process is presented in *Algorithm 4*.

Algorithm 4. Error Resilience Test

1. Initialize an empty *distances* list
2. For *i* to *N* repeat steps 3 - 8
3. Generate the original key for the protocol
4. Generate a faulty key for the protocol by changing noisiness parameter of the PUF (in the case of a lightweight cryptography protocol connection) or by changing one bit in the protocol key
5. Generate the expected response using the original key
6. Generate a response using the faulty key
7. Compare responses using Hamming Distance
8. Save the differences between responses in *distances* list
9. Calculate the average of *N* trials from the list of distances
10. Display information about the percentage of the average Hamming Distance for the function from *N* trials

Results

Table 1 presents the results of the Processing Time Test in milliseconds. The average times calculated over *N* trials remain stable in most cases, with fluctuations of around 1 ms, particularly in implementations involving PUF key generation. The largest variations are observed for the Kerberos protocol with a PUF-generated key, likely due to

its dual key-generation process. In standalone protocols, minor time differences are also observed as the number of samples increases.

Among the encryption protocols, a clear performance gap is visible between Ascon-128 and Speck. The execution time of Speck is more than three times shorter than Ascon's, which reflects their respective algorithmic complexity. When combined with PUF-generated keys, this relationship remains consistent. Generating ciphertext using Speck with a PUF key requires roughly half the time needed for PUF key generation alone, while Ascon with a PUF key takes almost the same time as generating the key itself.

For authentication protocols, the execution time difference between EAP and Kerberos reaches up to 0.04 ms. Kerberos requires two separate keys, which explains its higher runtime, especially when PUFs are involved. In contrast, EAP with a PUF-generated key shows timing results close to standalone key generation, indicating better efficiency.

Table 1: Processing Time Test for PUF key generation and authentication, stand-alone lightweight protocols and hybrid configurations (in ms) for 100, 200, 300, 400, 500 trials

	100	200	300	400	500
PUF Key Generation	34.6572	34.2931	34.1779	32.6941	32.9999
PUF Authentication	33.1938	34.6334	33.3981	31.7858	32.4303
Speck	0.0183	0.0091	0.0092	0.0121	0.0090
Ascon-128	0.3024	0.5018	0.4409	0.4035	0.3215
EAP	0.0086	0.0068	0.0039	0.0038	0.0098
Kerberos	0.0221	0.0176	0.0303	0.0440	0.0370
Speck with PUF	17.9759	16.7467	16.7978	17.2115	17.0284
Ascon-128 with PUF	34.9331	34.0792	34.2506	34.4157	34.4226
EAP with PUF	34.8274	35.4131	34.7048	32.6661	33.2669
Kerberos with PUF	68.2256	68.0962	65.8207	69.5025	65.8033

Table 2 shows the results of the **Memory Usage Test** in MiB. The lowest average memory consumption is observed for **Speck** (≈ 40 MiB) and **Kerberos** (≈ 44 MiB). Other standalone implementations, including **PUF Key Generation** and **PUF Authentication**, require up to 10 MiB more memory on average.

Across all protocols, the memory usage for hybrid versions (protocol + PUF) is nearly identical to that of standalone PUF key generation. This suggests that most of the memory overhead is related to the key generation process rather than to the cryptographic operation itself.

Table 2: Memory usage results for PUF key generation and authentication, stand-alone lightweight protocols and hybrid configurations (in MiB) for 100, 200, 300, 400, 500 trials

	100	200	300	400	500
PUF Key Generation	56.608	56.719	56.849	56.380	55.714
PUF Authentication	55.786	54.156	49.945	51.156	48.038
Speck	45.237	36.812	31.604	41.967	42.816
Ascon-128	54.680	53.660	53.442	54.367	53.270
EAP	54.526	53.999	54.145	50.944	43.246
Kerberos	45.231	45.830	45.572	41.910	41.994
Speck with PUF	56.924	57.105	57.113	56.580	56.272

Ascon-128 with PUF	57.098	57.055	56.785	56.144	56.033
EAP with PUF	56.554	56.197	56.592	56.499	56.653
Kerberos with PUF	56.567	56.083	56.881	56.493	57.068

Table 3 summarizes the outcomes of the **Attack Resilience Test** against machine-learning (ML) attacks. The results range between 0 and 1, where higher values indicate lower resistance. The following interpretation applies:

- > 0.7 - non resistant
- <= 0.7 - resistant

All implementations remain below the 0.7 threshold, confirming their **resilience to ML-based modeling attacks**. For hybrid configurations (protocols with PUF-generated keys), a minor increase in the resistance metric (~0.0020 units) is observed, implying a slightly higher susceptibility compared to standalone protocols.

Interestingly, **PUF key generation alone** achieves the highest resistance values, while hybrid configurations show marginally reduced resilience. No significant differences were observed between encryption and authentication categories.

Table 3: Attack Resilience results for PUF key generation and authentication, standalone lightweight protocols and hybrids configurations for 100, 200, 300, 400, 500 trials

	100	200	300	400	500
PUF Key Generation	0.5115	0.5159	0.5135	0.5134	0.5123
PUF Authentication	0.5038	0.5004	0.4988	0.4998	0.5001
Speck	0.4989	0.4987	0.4997	0.5007	0.4993
Ascon-128	0.4987	0.4994	0.4997	0.4991	0.5023
EAP	0.4981	0.5010	0.5008	0.5021	0.4973
Kerberos	0.4926	0.4977	0.5312	0.5306	0.5065
Speck with PUF	0.5016	0.5038	0.4973	0.4999	0.5026
Ascon-128 with PUF	0.5061	0.5006	0.4998	0.5022	0.4992
EAP with PUF	0.5115	0.5159	0.5135	0.5134	0.5123
Kerberos with PUF	0.5144	0.5111	0.5145	0.5130	0.5124

Finally, Table 4 presents the results of the **Error Resilience Test**, expressed as percentages and interpreted as follows:

- 100% - fully resistant
- 90-99% - highly resistant
- 70-89% - medium resistant
- <70% - non resistant

The PUF Key Generation and PUF Authentication implementations demonstrate high resilience (~93%), placing them in the “highly resistant” category. In contrast, Speck and EAP exhibit the lowest error resistance (<51%), even when combined with PUFs. Ascon-128 reaches the medium-resistance range (~70%), but its hybrid version falls below 50%, indicating reduced reliability under noise.

Kerberos, on the other hand, achieves nearly perfect resilience (≈ 99.6%) in standalone mode. However, when integrated with PUF key generation, its performance drops significantly, demonstrating the trade-off between enhanced hardware-based security and error tolerance.

Table 4: Error Resilience results for PUF key generation and authentication, standalone lightweight protocols and hybrids configurations (%) for 100, 200, 300, 400, 500 trials

	100	200	300	400	500
PUF Key Generation	93.57	93.63	93.68	93.76	93.82
PUF Authentication	93.71	93.77	93.83	93.89	93.84
Speck	47.84	50.14	50.83	50.05	50.26
Ascon-128	70.16	70.36	70.54	69.88	70.35
EAP	50.73	50.00	49.83	49.74	49.73
Kerberos	99.61	99.61	99.61	99.61	99.61
Speck with PUF	51.84	52.06	50.50	51.00	50.24
Ascon-128 with PUF	47.92	46.35	49.48	46.09	51.82
EAP with PUF	49.32	50.20	49.99	50.12	49.94
Kerberos with PUF	49.83	49.91	50.13	49.88	49.80

Conclusions

This study presents and tests a novel security architecture for IIoT systems that combines Lightweight Cryptographic Protocols with Physically Unclonable Functions (PUF) as a mechanism for key generation and device authentication. The aim was to evaluate whether combining Lightweight Cryptographic Protocols with Physically Unclonable Functions (PUF) is a feasible solution and test it in terms of overall usefulness, processing time, energy consumption, resistance to attacks and error resilience.

The experimental results confirm that integrating PUFs enhances key uniqueness and resistance to modelling attacks, as proofed by the Attack Resilience Test. However, this improvement comes at the cost of lower resistance to transmission errors, demonstrated by the Error Resilience Test results.

Kerberos in its standalone version proved to be the most resistant protocol in terms of error resilience, reaching >99% resistance. However, its hybrid version while combined with PUF, despite high security, did not achieve similar results in resistance to errors.

Speck, despite good time efficiency and low memory consumption, showed low resistance to errors, which may limit its use in unstable industrial environments. Ascon-128 achieved balanced results in most tests, but the version with PUF did not exceed the threshold of 50% resistance to errors, which disqualifies it in applications requiring reliability.

Simple PUF functions (PUF Key Generation and PUF Authentication) without additional cryptographic layers showed the highest resistance to both errors and attacks, but do not offer full encryption or access control mechanisms.

Combining lightweight cryptography with PUF is feasible and brings significant security benefits, especially in the context of unique device identity and making it difficult to intercept keys. However, this requires a compromise

between security and resistance to interference, especially when devices are deployed in an environment with a high level of noise or power instability.

As for the future work on combining Lightweight Cryptographic Protocols with Physically Unclonable Functions, several research directions can be pursued. Firstly, it would be valuable to test the proposed hybrid solution on real IIoT devices and in industrial environments, in order to assess performance in terms of processing time, memory usage and resilience to errors and Machine Learning attacks under realistic conditions. Secondly, further investigation should focus on whether other cryptographic security protocols can be integrated with PUF based key generation and how they perform in the tests applied in this work. Additionally, resilience to other types of attacks - beyond those based on machine learning - should be examined, with a strong emphasis on testing in real-life industrial environments. Finally, a comparative analysis between the hybrid solution presented in this thesis and alternative approaches, such as blockchain combined with PUF, could provide a broader insights into the effectiveness and practicality of different PUF based architectures for industrial security.

Bibligraphy

- Babaei, A., Schiel, G. (2019), “Physical Unclonable Functions in the Internet of Things: State of the Art and Open Challenges”, Sensors, 19, 3208
- Wang, M., Sun, Y., Sun, H., Zhang, B. (2023), “Security Issues on Industrial Internet of Things: Overview and Challenges”, Computers, 12, 12
- Alotaibi, B. (2023), “A Survey on Industrial Internet of Things Security: Requirements, Attacks, AI-Based Solutions, and Edge Computing Opportunities”, Sensors, 23, 7470
- Tan, S.F., Samsudin, A. (2021), “Recent Technologies, Security Countermeasure and Ongoing Challenges of Industrial Internet of Things (IIoT): A Survey”, Sensors, 21, 6647
- Khan, R., Eldeeb, H. B., Mefgouda, B., Alhussein, O., Saleh, H., Muhaidat, S. (2025), “Encoder decoderbased Virtual Physically Unclonable Function for Internet of Things device authentication using splitlearning”, Computers & Security, 148
- Bathalapalli, V. K. V. V., Mohanty, S. P., Pan, C., Kougianos, E. (2023), “QPUF: Quantum Physical Unclonable Functions for Security-by-Design of Industrial Internet-of-Things” 2023 IEEE International Symposium on Smart Electronic Systems (iSES), Ahmedabad, India, 296-301
- Wang, B., Cui, Y., Gu, C., Wang, C., Liu, W. (2023), “Novel Intrinsic Physical Unclonable Function Design for Post-quantum Cryptography”, Proceedings of the 2023 IEEE International Symposium on Circuits and Systems (ISCAS)
- Mishra, N., Islam, S. K. H., Zeadally, S. (2024), “A survey on security and cryptographic perspective of Industrial-Internet-of-Things”, Internet of Things, 25
- Cultice, T., Thapliyal, H. (2022), “PUF-Based Post-Quantum CAN-FD Framework for Vehicular Security”, Information, 13, 382
- Sengupta, J., Ruj, S., Sipra Das Bit, S. D. (2020), “A Comprehensive Survey on Attacks, Security Issues and Blockchain Solutions for IoT and IIoT”, Journal of Network and Computer Applications, 149
- Aghapour, S., Ahmadi, K., Anastasova, M., Mozaffari Kermani, M.,Azarderakhsh, R. (2024), "PUFKyber: Design of a PUF-Based Kyber Architecture Benchmarked on Diverse ARM Processors," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 43, 12, 4453-4462
- Fan C. -I., Lai C. -I., Vishwasrao Medhane D. (2024), "CAKE-PUF: A Collaborative Authentication and Key Exchange Protocol Based on Physically Unclonable Functions for Industrial Internet of Things", in IEEE Internet of Things Journal, 11, 24, 39709-39720
- Dąbrowska, A. (2024), „Security in Industry 5.0: Physically Unclonable Functions in Resource Constrained Devices”, Engineering diploma thesis
- https://pypuf.readthedocs.io/en/latest/
- https://pypi.org/project/pypuf/0.0.3/