# Inter-Device PUFs: A Novel Paradigm for Physical Unclonable Functions

Emiliia Geloczi and Stefan Katzenbeisser

University of Passau, Innstr. 43, 94036 Passau, Germany
{emiliia.geloczi,stefan.katzenbeisser}@uni-passau.de

**Abstract.** A Physical Unclonable Function (PUF) is a promising concept for device authentication, as it can be extracted from existing device components and enables the generation of unique, device-specific keys. However, existing PUF-based approaches often require prior distribution of these keys during an enrolment phase, require persistent storage or transmission of sensitive data, and are vulnerable to different attacks. To address these limitations, we propose a novel approach to using PUFs. While PUFs are typically employed by leveraging the inherent uniqueness of their responses, our approach instead focuses on their similarities, exploiting matching segments of two distinct PUFs. This enables the identification of a PUF that is unique per-device-pair, termed Inter-Device PUF (ID-PUF), which links two devices and serves as the basis for a shared secret key. This key can be independently generated on both devices, eliminating the need for its prior distribution and persistent storage, thus making an ID-PUF well-suited for application in lightweight symmetric cryptographic schemes. The feasibility of an ID-PUF is demonstrated through statistical analysis and experimental evaluation on several devices. Furthermore, to illustrate its application, we present a novel ID-PUF-based mutual authentication protocol (ID-PUMA) and analyse its security.

**Keywords:** PUF · ID-PUF · Mutual Authentication · ID-PUMA

## 1 Introduction

The number of Internet of Things (IoT) devices is rapidly increasing, currently estimated at around 23.15 billion and projected to reach 39.65 billion by 2028 [40]. During operation, devices generate, process and transfer large amounts of data, which can be critical in certain contexts, becoming attractive targets for adversaries [30]. Despite many existing security measures [31], the rise in newly discovered vulnerabilities and attacks [11], combined with the heterogeneous and resource-constrained nature of IoT systems, requires the continuous evolution of security solutions [45].

In order to secure an IoT system, solutions should not only guarantee compliance with the fundamental security principles known as the CIA Triad (confidentiality, integrity, and availability) [39], but also ensure privacy, auditability,

This is a Pre-print: The final version will appear in the proceedings of Computer Security – ESORICS Workshops 2025 by Springer Nature.

non-repudiation, and trust [25]. In particular, authentication plays a critical role in IoT [28], as it establishes mutual trust between system parties and, hence, facilitates many security properties, thereby reducing overall security risks [2].

For lightweight devices, Physical Unclonable Functions (PUFs) have been proposed as a promising means of authentication. A PUF is an intrinsic device fingerprint [21], unique for each device and unclonable due to minor deviations of the hardware components [1], which cannot exactly be reproduced, even by the manufacturer. PUFs can be extracted from components already present in a device, such as memory modules [8], thereby eliminating the need for additional hardware. Moreover, they can serve as the basis for device-specific keys that are generated on demand, thus removing the requirement for persistent key storage. Despite these advantages, existing PUF-based authentication solutions often rely on a prior key exchange during the enrolment phase, require persistent storage or transmission of sensitive data, and remain vulnerable to known attacks [28].

In this paper, we propose a novel approach to applying PUFs that addresses the listed limitations while preserving the advantages of PUF-based security. Traditional PUF-based solutions typically rely on the uniqueness of PUF responses to distinguish between devices and verify their identities. In contrast, we explore the potential of leveraging matching segments of PUFs from two devices to form a unique characteristic related to both, allowing their inter-device association. We name this characteristic an Inter-Device PUF (ID-PUF). An ID-PUF enables the generation of a shared key that is unique for the device pair, thus facilitating the use of lightweight symmetric cryptography. Since this key can be independently reproduced on both devices on demand, there is no need for it to be distributed during enrolment. To demonstrate the possible application of ID-PUFs, we also introduce a novel mutual authentication protocol based on this concept. The results of our statistical and security analyses confirm the feasibility and advantages of using ID-PUFs as a foundation for authentication solutions, particularly in terms of resistance to common attacks.

In summary, we present the following two key **contributions**:

- *ID-PUF:* A novel paradigm for PUFs that leverages matching segments from two different PUFs to generate a shared key that is unique per-device-pair. It preserves PUF advantages while addressing common limitations of traditional PUF-based solutions, making it a robust foundation for symmetric cryptographic schemes and mutual device authentication.
- *Mutual Authentication Protocol:* A novel lightweight ID-PUF-based protocol demonstrating the concept's applicability and enhanced security over existing solutions.

The rest of the paper is organised as follows. Section 2 provides general background information on PUFs. Section 3 introduces the proposed ID-PUF concept. The mutual authentication protocol is described in Section 4, followed by its security analysis in Section 5. Section 6 offers an overview of related work and their comparison with our approach. In Section 7, we discuss potential challenges in the application of ID-PUFs. Finally, Section 8 concludes the paper and outlines directions for future research.

This is a Pre-print: The final version will appear in the proceedings of Computer Security – ESORICS Workshops 2025 by Springer Nature.

## 2 Background: Physical Unclonable Functions

During the manufacturing of electronic devices, variations in the characteristics of their components naturally occur. Even devices originating from the same production series exhibit slight differences in their physical properties [28]. While these variations do not impact functionality or reliability, they can be used to create a unique fingerprint for a device. Such a fingerprint is known as a Physical Unclonable Function (PUF), a function embedded in a physical object that receives a challenge $c$ and generates a response $r$, forming a challenge-response pair [21]. Ideally, PUF should exhibit the following properties [18]:

- *Unclonability:* A PUF should not be physically reproducible or clonable, even by its manufacturer.
- *Unpredictability*: It should be impossible to predict the response $r$ for a new challenge $c$, even if other challenges and corresponding responses have been previously observed.
- *Reproducibility:* The same challenge $c$ should consistently produce the same response $r$ on the same physical object.
- *Tamper-Evidence*: Any physical tampering with a PUF should significantly alter its response.

PUFs can be extracted from various device components and characteristics, such as memory modules or signal timing patterns [8]. In our work, we use a start-up Static Random Access Memory (SRAM) PUF [19], which exhibits a limited number of challenge-response pairs [38], demonstrating that our approach can also be applied to so-called "weak" PUFs. Additionally, SRAM is present in most devices, not easily accessible externally, and relatively stable [44].

Due to slight variations in transistor properties at power-up, SRAM modules generate bit patterns that appear random but are stable and unique to each module. These patterns can serve as a distinct fingerprint of a device. In the case of SRAM PUFs, a memory address serves as a challenge $c$ while its corresponding uninitialised value, read out immediately after power-up, represents a response $r$. Multiple challenge-response pairs constitute the challenge set $C := \{c_i\}$ and the corresponding response set $R := \{r_i\}$ $(\forall i = 0, \ldots, n)$, together forming the set of challenge-response pairs $CRP := (C, R)$. It is important to note that the number of challenge-response pairs $(n)$ may vary depending on the application requirements. Furthermore, a single PUF can produce many distinct $CRPs$.

## 3 Inter-Device PUF

Due to the uniqueness of PUF responses [24], device-specific cryptographic keys and robust device identification can be achieved. Typically, $CRPs$ from a PUF are collected during an enrolment process, and one of them is later used for authentication or key generation. We propose an alternative to this canonical use of PUFs. Instead of relying on a single device's uniqueness, our approach leverages matching response segments from the PUFs of two devices to identify a device-pair-specific PUF, termed an Inter-Device PUF (ID-PUF), which enables inter-device association.

This is a Pre-print: The final version will appear in the proceedings of Computer Security – ESORICS Workshops 2025 by Springer Nature.

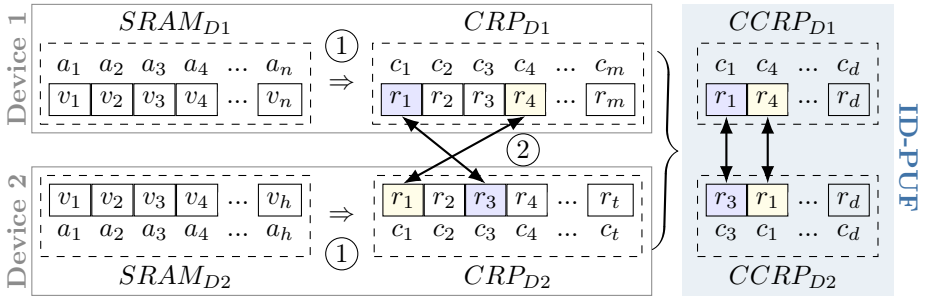



Fig. 1: Identification procedure of SRAM-based ID-PUF.

### 3.1 ID-PUF Identification

In this section, we detail an ID-PUF identification process, consisting of two phases: ① PUF Initialisation and ② Collision Search (see Figure 1).

**① PUF Initialisation**

Consider two devices $D1$ and $D2$, each equipped with an SRAM module. These modules contain $n$ and $h$ addresses ($a$) with corresponding values ($v$), respectively. Hence, the total memory available on each device is $SRAM_{D1} := \{(a_i, v_i) | \forall i = 1, \ldots, n\}$ and $SRAM_{D2} := \{(a_j, v_j) | \forall j = 1, \ldots, h\}$.

The first phase is dedicated to selecting combinations $(a, v)$ that are suitable to serve as a PUF. Their number depends on the size of the SRAM module, its stability, and PUF-metric thresholds applied during selection. These PUF-suitable combinations form the device-specific $CRP$, where $a$ corresponds to the challenge $c$, and $v$ corresponds to the response $r$, i.e., $(a, v) \rightarrow (c, r)$. In our case, this results in two sets: $CRP_{D1} := \{(c_i, r_i) | \forall i = 0, \ldots, m\}$ and $CRP_{D2} := \{(c_j, r_j) | \forall j = 0, \ldots, t\}$. To identify these $CRPs$, all $(a, v)$ combinations unsuitable for use as a PUF are filtered out. This requires evaluation of potential responses ($v$), which can be done using the following PUF metrics [24]:

- The *uniformity* of $v$ is evaluated using the fractional Hamming Weight $HW(v)$. The closer $HW(v)$ is to the ideal value of 0.5, the more uniform $v$ is.
- The *reliability* of $v$ is evaluated using the fractional intra-device Hamming Distance $HD_{intra}(v, v')$, which reflects the difference between multiple responses to the same corresponding challenge on a single device. A lower $HD_{intra}$ indicates a more stable response; hence, the ideal value is 0.
- The *uniqueness* of $v$ is evaluated using fractional inter-device Hamming Distance $HD_{inter}(v_1, v_2)$, which measures the difference between responses to the same challenge across different memory modules, indicating the ability of a PUF to produce unique responses and ideally tends to 0.5.

Thus, combinations $(a, v)$ that eventually form the $CRPs$ and become $(c, r)$ are selected such that the corresponding $r$ values meet predefined thresholds for each metric.

**② Collision Search**
After performing PUF initialisation, we obtain two sets $CRP_{D1}$ and $CRP_{D2}$ that are used to further identify Collision $CRPs$ ($CCRPs$) for the device pair $D1$ and $D2$. To achieve this, we need to select pairs $(c_i, r_i) \in CRP_{D1}$ and $(c_j, r_j) \in CRP_{D2}$, where collisions of responses occur, i.e., $r_i = r_j$ and $c_i \neq c_j$, thereby forming $CCRP_{D1}$ and $CCRP_{D2}$ related to the ID-PUF of the devices:

$$CCRP_{D1} := \{(c_i, r_i)\},\ CCRP_{D2} := \{(c_j, r_j)\},\ \forall i, j = 0, \ldots, d.$$

For each $CCRP$, the sets of included challenges $c$ and their corresponding responses $r$ are referred to as the collision challenge set $CC$ and the collision response set $CR$, respectively. Thus, for devices $D1$ and $D2$:

$$CCRP_{D1} := \{CC_{D1}, CR_{D1}\},\ CCRP_{D2} := \{CC_{D2}, CR_{D2}\}.$$

Given that $CR_{D1} = CR_{D2} = CR$, we can rewrite $CCRPs$ as:

$$CCRP_{D1} := \{CC_{D1}, CR\},\ CCRP_{D2} := \{CC_{D2}, CR\}.$$

Both identified $CCRPs$ contain an identical set of responses $CR$ that can be used to generate a shared key for the two devices. However, using the entire set at once would allow for the creation of only a single key for the device pair. In practice, though, there may be a need to generate multiple keys, e.g., to support multi-channel communication. To address this, various subsets of $CR$ can be selected for key generation, enabling the creation of multiple unique keys.

### 3.2 Statistical Analysis

The uniqueness property seems to contradict the concept of an ID-PUF. Still, in this section, we demonstrate that, in theory, a sufficient number of response collisions can be found in SRAM PUFs of realistic size to enable key generation.

Assume that there are two devices $D1$ and $D2$ with corresponding $CRP$ sets of sizes $m$ and $t$, identified in ① PUF initialisation step:

$$CRP_{D1} := \{(c_i, r_i) | \forall i = 0, \ldots, m\},\ CRP_{D2} := \{(c_j, r_j) | \forall j = 0, \ldots, t\}.$$

Due to uniformity, a response $r$ is a $w$-bit random-looking value drawn from a discrete set of $2^w$ possible values. Responses are independent and may recur within the same $CRP$.

*Probability of Collision Occurrence.* To calculate the probability $P(COL)$ that at least one response collision ($r_i = r_j$, $c_i \neq c_j$) occurs between $CRP_{D1}$ and $CRP_{D2}$, we apply the complementary event approach [9]. Thus, $P(COL)$ can be expressed as 1 minus the probability that no response collision between two $CRPs$ exists. Additionally, in our case, $m \times t \gg 2^w$, indicating that the values involved are relatively large. We can then employ the first-order exponential approximation $(1 - x)^y \approx e^{-xy}$ [4]. Consequently, the probability of a collision can be approximated by

$$P(COL) \approx 1 - e^{-\frac{mt}{2^w}}. \tag{1}$$

Therefore, to expect collisions, the probability $P(COL)$ must approach 1.

*Expected Number of Collisions.* To calculate the expected number of collisions $E[X]$, we first define an indicator random variable $X_{i,j}$ for each pair of responses $r_i \in CRP_{D1}$ and $r_j \in CRP_{D2}$ [37]:

$$X_{i,j} = \begin{cases} 1, & \text{if } r_i = r_j, \\ 0, & \text{if } r_i \neq r_j. \end{cases}$$

The total number of collisions is then given by $X = \sum_{i=1}^{m} \sum_{j=1}^{t} X_{i,j}$. Finally, by applying the linearity of expectation [13], we can express $E[X]$ as:

$$E[X] = \sum_{i=1}^{m} \sum_{j=1}^{t} E[X_{i,j}].$$

Since $X_{i,j} = 1$ with the probability that a response $r_i \in CRP_{D1}$ matches a specific response $r_j \in CRP_{D2}$, we have $E[X_{i,j}] = \frac{1}{2^w}$. Thus, the total expected number of collisions can be calculated using the formula:

$$E[X] = \sum_{i=1}^{m} \sum_{j=1}^{t} \frac{1}{2^w} = \frac{mt}{2^w}. \tag{2}$$

*Number of Keys.* As mentioned in Section 3.1, using different subsets of the collision response set $CR := \{r_i \mid \forall i = 0, \ldots, d\}$, which is identical for both devices, enables the generation of multiple distinct keys for a single device pair. Hence, to estimate the number of keys of varying lengths that can be generated from $CR$, we count the total number of unique subsets of $CR$ of a specified length, taking into account the order of the included responses $r$. For clarity, we refer to this number of subsets as the number of potential keys. The calculations use the formulas for permutations with ($NK_{WR}$) and without ($NK_{NR}$) repetitions [36], depending on whether repeated responses are allowed:

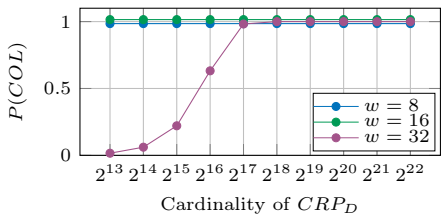
$$NK_{WR} = P'(d, l) = d^l, \quad NK_{NR} = P(d, l) = \frac{d!}{(d-l)!}, \tag{3}$$

where $d$ is the cardinality of the collision response set $CR$, and $l$ is the number of $w$-bit responses needed to construct a key of a length $k$, i.e., $l \times w = k$.

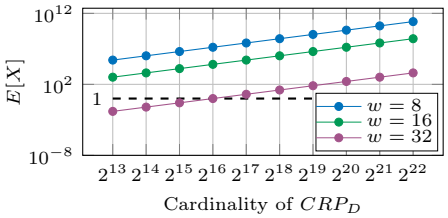
*Estimation Results.* Figure 2 shows estimations of $P(COL)$, $E[X]$, $NK_{NR}$ and $NK_{WR}$ for various response lengths $w$ (8, 16 and 32 bit) and different cardinalities of $CRPs$ from two devices. For simplicity, we assume that $|CRP_{D1}| = |CRP_{D2}| = |CRP_D|$, i.e., $CRPs$ consist of the same number of $(c, r)$ pairs (e.g., $2^{13}, \ldots, 2^{22}$).

As shown in Figure 2a, collisions are guaranteed to happen for relatively short response lengths. However, this is not the case for 32-bit responses, where the probability approaches 1 only when more than $2^{17}$ responses are available. Similarly, Figure 2b shows that the expected number of collisions is relatively high for 8 and 16-bit response lengths; in contrast, for 32-bit responses, the expected number of collisions is significantly lower, often less than one, meaning that in some cases, no collisions are expected at all.

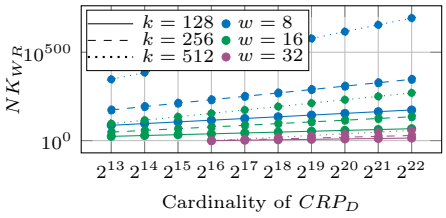
Based on the estimated expected number of collisions, we then calculated the number of distinct keys of varying lengths ($k$) that can potentially be generated. Figure 2c shows the number of keys that allow repetitive responses. It can be



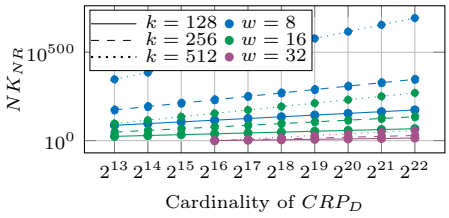

(a) Probability of collision occurrence.




(b) Expected number of collisions.




(c) Number of possible distinct keys allowing repetitions of $r$.




(d) Number of possible distinct keys without repetitions of $r$.

Fig. 2: Theoretical estimations of $P(COL)$, $E[X]$, $NK_{WR}$ and $NK_{NR}$; $w$ denotes the response length (in bits), $k$ denotes the key length (in bits).

observed that a large number of keys of all considered lengths can be constructed from 8- and 16-bit responses. However, in the case of 32-bit responses, where fewer collisions are expected, keys can only be generated if the cardinalities of $CRPs$ are at least $2^{16}$. Similar results are observed for keys that contain no repeated responses (see Figure 2d). Although their total number is slightly lower than that of keys allowing repetition, it remains large enough that the difference is negligible given the magnitude of the values.

The results of our statistical analysis indicate that it is theoretically possible to identify ID-PUFs for device pairs, even when the devices have short PUF segments, i.e., relatively limited SRAM.

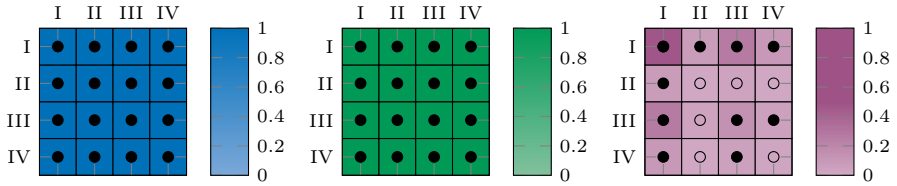
### 3.3 Experimental Implementation

To validate the estimates presented in the previous section in practice, we performed the ID-PUF identification on four existing development boards listed in Table 1. Since five instances of each board type were used, the results in this section represent average values. For instance, when analysing the probability of collisions between boards I and II, the reported value represents the average across the results obtained for all 25 possible instance combinations ($5 \times 5$).

During ① PUF initialisation, we selected $(a, v)$ that satisfied the following conditions: $HW(v), HW_{inter}(v_1, v_2) \in [0.4; 0.6]$ and $HD_{intra}(v, v') \leq 10\%$. As a result, on average, 50% of each device's memory was identified as PUF-suitable and was used for ② collision search (see Table 1). Since all boards support 8/16/32-bit word sizes, collisions were searched for across these response lengths.

Table 1: Development boards selected for the ID-PUF identification.

| № | Board's Name | SRAM | | Amount |
|---|---|---|---|---|
| | | Initial | PUF-suitable | |
| I | ESP32-S3 [14] | 416 kB | 188 kB | 5 |
| II | ESP8266EX [15] | 50 kB | 23 kB | 5 |
| III | Arduino Nano 33 (internal SRAM) [3] | 200 kB | 102 kB | 5 |
| IV | Arduino Nano 33 (external SRAM) [3,29] | 512 kb | 266 kb | 5 |

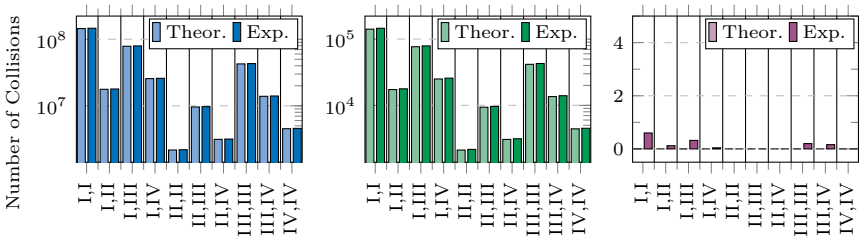



(a) 8-bit response (b) 16-bit response (c) 32-bit response

Fig. 3: Theoretical probability of collision occurrence (heat map) and experimental validation (● – at least one collision is observed, ○ – no collisions were observed) between ESP32-S3 (I), ESP8266EX (II), and Arduino Nano 33 with internal (III) and external SRAM (IV).

*Probability of Collision Occurrence.* First, we used Equation 1 to theoretically calculate the probability of collision occurrence $P(COL)$ for all combinations of the selected boards. The results are presented in Figure 3 as a heat map. Additionally, in order to reflect experimental findings, we marked the board combinations where at least one collision was observed with ●, and if no collisions were found with ○. It can be seen that collisions are theoretically guaranteed, and confirmed experimentally, for 8- and 16-bit responses. In contrast, for 32-bit responses, the probabilities are significantly lower due to limited memory on the boards. Nevertheless, despite the low theoretical probability, some collisions were still observed in practice.

*Expected Number of Collisions.* First, we calculated the expected number of potential collisions using Equation 2. Then, we experimentally tested our boards to verify whether the observed number of collisions aligns with the theoretical prediction. The results, presented in Figure 4, show that the number of detected collisions closely matches the theoretical assessment. Additionally, it is visible that the number of collisions for 8- and 16-bit responses is significantly higher than for 32-bit responses. In the case of 32-bit responses, the theoretical prediction indicates no collisions; however, in practice, a few collisions were still observed among board combinations with the largest available memory. As illustrated in Figure 4c, the average number of detected collisions per combination remains below one, indicating that, despite the theoretical prediction, one or two collisions occasionally occurred.




(a) 8-bit response (b) 16-bit response (c) 32-bit response

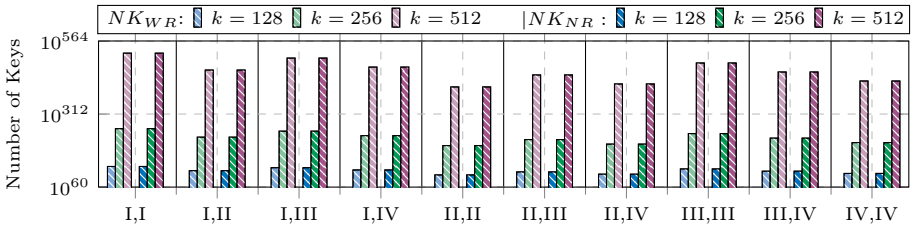
Fig. 4: Number of collisions, both theoretically predicted and experimentally detected, between ESP32-S3 (I), ESP8266EX (II), and Arduino Nano 33 with internal (III) and external SRAM (IV).

*Number of Keys.* As mentioned earlier in Section 3.2, to determine the number of distinct keys that can be generated from the collision response set $CR$, which is shared by both devices, we count the number of possible subsets of $CR$, as each subset serves as a seed for one cryptographic key. Additionally, we assume that each subset used for key generation has the same length as the resulting key. Figure 5 presents the results of our assessment of the potential number of distinct keys. For the theoretical calculations, we used the expected number of collisions as the cardinality of $CR$ while for the real-world assessment, we used the actual number of detected collisions (see Figure 4). Due to an insufficient number of collisions for 32-bit responses, the results are presented only for 8- and 16-bit responses.

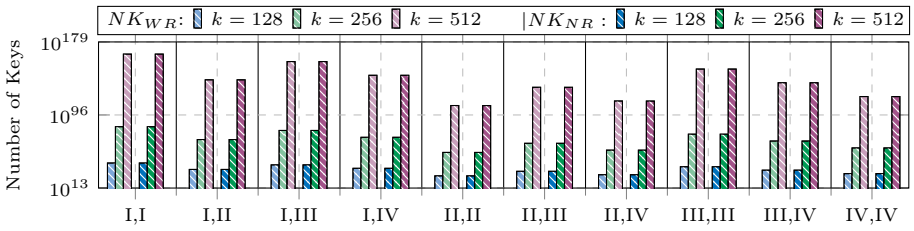
In our experiment, we count the number of keys of lengths $k$ equal to 128, 256, and 512 bits under two scenarios: with ($NK_{WR}$) and without ($NK_{NR}$) including repeated responses (see Equation 3). The results show that the theoretical predictions closely match the actual observed values, appearing nearly identical given the magnitude of the values. Regarding the difference between $NK_{WR}$ and $NK_{NR}$, we can see that although $NK_{WR}$ is consistently higher than $NK_{NR}$, it is almost indistinguishable due to the scale of the values. Overall, the potential number of distinct keys is extremely large. Even for boards with the smallest available memory (e.g., II, II), it exceeds $10^{14}$.

The results of the statistical analysis and experimental implementation demonstrate that the existence of collisions, and thus the feasibility of ID-PUFs, can be guaranteed, even when only relatively limited PUF-suitable memory is available. For example, using Equation 1, it can be calculated that to ensure the presence of collisions ($P(COL) \approx 1$), each device must have at least 280 bit/8.6 kb/550 kB of PUF-suitable memory for 8/16/32-bit words, respectively, which is significantly less than what is typically available. Furthermore, the quantity of collisions is sufficient to generate thousands of distinct keys of any recommended lengths for both symmetric (from 128-bit) and asymmetric (to 4096-bit) encryption schemes [16,32].

This is a Pre-print: The final version will appear in the proceedings of Computer Security – ESORICS Workshops 2025 by Springer Nature. 




(a) 8-bit response




(b) 16-bit response

Fig. 5: Number of distinct keys of various lengths ($k$-bit) that can be generated from theoretically estimated (colored) and experimentally observed (patterned) collisions between ESP32-S3 (I), ESP8266EX (II), and Arduino Nano 33 with internal (III) and external SRAM (IV).

## 4 Mutual Authentication Protocol Based on ID-PUF

In this section, we present a novel ID-PUF-based Mutual Authentication protocol (ID-PUMA) that enables an inter-device association mechanism without the need for a prior distribution of a secret key, its persistent storage, and transmission of sensitive data.

ID-PUMA involves three parties: two devices ($D1$ and $D2$), each equipped with SRAM, and a manager $M$ that performs ID-PUF identification, including PUF initialisation on both devices and consequent collision search (see Section 3.1). If both devices originate from the same manufacturer, it serves as $M$; otherwise, a user-side application may fulfil this role.

In describing the protocol workflow, we do not revisit the ID-PUF identification procedure, as it has already been detailed in Section 3.1. We assume that the manager $M$ has already completed this process: derived a shared collision response $CR$ and the corresponding collision challenges $CC_{D1}$ and $CC_{D2}$, and embedded these challenges into the respective devices. While the challenges are public, the response $CR$ is considered secret and is never persistently stored or transmitted between system parties. Additionally, $M$ distributes the identifiers of the devices involved in the protocol, $ID_{D1}$ and $ID_{D2}$. The symbols used in the protocol are listed in Table 2, followed by a detailed explanation of the workflow that is also illustrated as a sequence diagram in Figure 6.

Table 2: Symbols used for the protocol description.

| Symbol | Description |
|---|---|
| $D1$, $D2$ | Devices participating in mutual authentication |
| $M$ | Manager responsible for ID-PUF identification |
| $ID_{D1}$, $ID_{D1}$ | Unique identifiers of the devices $D1$ and $D2$ |
| $CC_{D1}$, $CC_{D2}$ | Collision challenges embedded in the corresponding devices |
| $CR$ | Collision response derived from devices' SRAMs using $CC_{D1}$ or $CC_{D2}$ |
| $SK$ | Secret shared key generated on demand using $CR_x$ |
| $N_{D1}$, $N_{D2}$ | Nonces generated by the devices |
| $Encr_{SK}(\ldots)/Decr_{SK}(\ldots)$ | Encryption/decryption functions using $SK$ |
| $E_{SK}(\ldots)$ | Value encrypted using $SK$ |
| $FindCC(ID_x)$ | Function to retrieve the $CC$ associated with $ID_x$ |
| $DeriveCR(CC_x)$ | Function to derive $CR$ from $CC_{D1}$ or $CC_{D2}$ |
| $KeyGen(CR_x)$ | Function to generate a secret key $SK$ based on $CR$ |
| $GenerateNonce()$ | Function to generate a random nonce |
| $Hash(\ldots)$ | One-way hash function used to create a hash value |
| $H(\ldots)$ | Hash value produced using the function $Hash(\ldots)$ |

Assume that $D1$ initiates mutual authentication with $D2$. In the first step (1), $D1$ identifies $CC_{D1}$ associated with $D2$ using $ID_{D2}$. Based on $CC_{D1}$, $D1$ then derives the corresponding $CR$ that is used to generate a shared secret key $SK$. Next, $D1$ generates a random nonce $N_{D1}$ and encrypts it using $SK$, resulting in $E_{SK}(N_{D1})$. Finally, $D1$ computes $H_1(ID_{D2}, N_{D1})$. In the second step (2), $D1$ sends to $D2$ its identifier $ID_{D1}$, the previously computed $H_1(ID_{D2}, N_{D1})$, and the encrypted nonce $E_{SK}(N_{D1})$. Upon receiving the message from $D1$, $D2$ identifies $CC_{D2}$ associated with $D1$ using $ID_{D1}$, derives $CR$ corresponding to $CC_{D2}$ and generates the shared secret key $SK$. Using this $SK$, $D2$ decrypts the nonce received from $D1$ and obtains $N'_{D1}$ (3).

Next, the first comparison step, denoted as $(C.I)$, is performed. $D2$ computes the hash value $H'_1$ of its identifier $ID_{D2}$ and the decrypted nonce $N'_{D1}$. The computed hash value is then compared with the hash $H_1$ received from $D1$. To proceed with the mutual authentication process, these values must match; otherwise, the authentication fails and is terminated.

If authentication proceeds, $D2$ initiates the next step (4). First, $D2$ generates a nonce $N_{D2}$ and encrypts it using $SK$ generated during step (3). Then, $D2$ computes a hash value $H_2$ based on both device identifiers $ID_{D1}$ and $ID_{D2}$, as well as the two nonces: the decrypted nonce $N'_{D1}$ received from $D1$ in step (3), and the previously generated $N_{D2}$. Finally, $D2$ sends $H_2$ and the encrypted nonce $E_{SK}(N_{D2})$ to $D1$ (5). Upon receiving the message from $D2$, $D1$ proceed to step (6), where similarity to step (1), $D1$ generates the shared secret key $SK$ and uses it to decrypt $E_{SK}(N_{D2})$, obtaining $N'_{D2}$.

Next, $D1$ performs the second comparison step $(C.II)$. It computes the hash value $H'_2$ based on both device identifiers $ID_{D1}$ and $ID_{D2}$, as well as the two nonces: $N_{D1}$, generated in step (1), and $N'_{D2}$, obtained in step (6). This $H'_2$ is then compared with the value $H_2$ received from $D2$. If these values do not match, the authentication is terminated; otherwise, the authentication is successful.

It is important to note that $SK$ and $CR$ do not need to be stored permanently in the device memory, as they are derived on demand and can be deleted immediately after use.

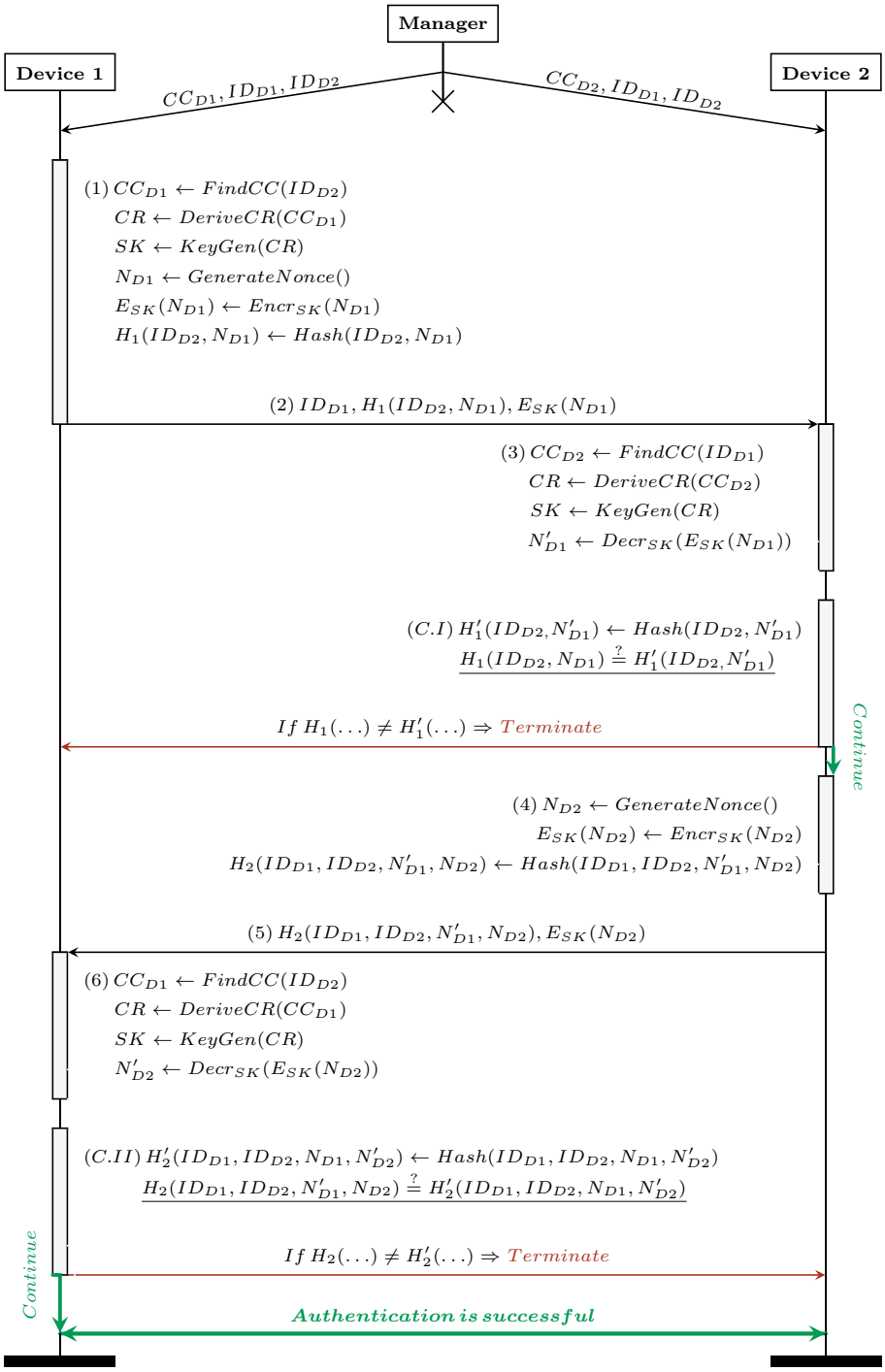



Fig. 6: Workflow of the proposed ID-PUF-based mutual authentication protocol.

# 5 Security Analysis

In this section, we perform a security analysis of the proposed mutual authentication protocol ID-PUMA. First, we describe the system and adversary models, then we discuss the feasibility of potential attacks against the protocol.

## 5.1 System Model

The system consists of legitimate and trusted protocol parties (devices $D1$ and $D2$, a manager $M$) and an adversary $\mathcal{A}$. $D1$ and $D2$ are engaged in the mutual authentication protocol, in which $M$ is not directly involved. Instead, $M$ is responsible solely for the secure preparatory phase that includes ID-PUF identification and the distribution of the associated collision challenges ($CCs$) and device identifiers ($IDs$) used later for secret key generation. In addition, we assume that only strong cryptographic schemes, primitives, or random number generation functions are employed within the system [7].

## 5.2 Adversary Model

The adversary $\mathcal{A}$ aims to disrupt the mutual authentication between $D1$ and $D2$ by performing attacks within their capabilities and limitations. We adapt the Dolev-Yao model [12], under which $\mathcal{A}$ has full control over the communication between $D1$ and $D2$. Accordingly, $\mathcal{A}$'s capabilities include intercepting, injecting, and manipulating any transmitted message; establishing a connection with any party; and impersonating any party within the system. However, $\mathcal{A}$ has limited capabilities regarding cryptographic primitives and, therefore, cannot perform the following actions: guess or generate nonces identical to legitimate ones; correctly decrypt or encrypt messages without possessing the secret key $SK$; break or forge hash values. Additionally, we assume that $\mathcal{A}$ does not collaborate with any of the protocol parties to deceive the other one, nor with $M$ to obtain any information, e.g., SRAM or ID-PUF.

## 5.3 Analysis of the Protocol's Resistance to Attacks

In the following, we analyse the resistance of the proposed protocol (see Section 4) to nine common attacks [23,28], based on the system and adversary models introduced in Sections 5.1 and 5.2.

*Eavesdropping Attack:* $\mathcal{A}$ eavesdrops on communication between $D1$ and $D2$ to extract information for future attacks. In our protocol, $\mathcal{A}$ can capture $ID_{D1}\,(2)$, which is public; encrypted nonces $E_{SK}(N_x)\,(2,5)$, which cannot be decrypted without $SK$; and $H_x(\ldots)\,(2,5)$, which $\mathcal{A}$ cannot break. Furthermore, none of these items provides sufficient information to facilitate other attacks.

*Replay Attacks:* $\mathcal{A}$ intercepts and retransmits messages in an attempt to impersonate $D1$ or $D2$. Although the intercepted $ID_{D1}$ can be reused, successful authentication requires encrypting a fresh nonce with $SK$, which $\mathcal{A}$ cannot perform. Consequently, the attack fails when $D2$ attempts to decrypt $E_{SK}(N_{D1})$ in step (3) or verify the hash in step ($C.I$). Reusing of encrypted nonces or hashes would also fail due to nonce freshness.

*Man-in-the-Middle (MitM) Attack:* $\mathcal{A}$ intervenes the communication, causing the devices interact with $\mathcal{A}$ instead of each other. During attack in step (2)/(5), $\mathcal{A}$ can replace the legitimate $N_x$ with $N_{\mathcal{A}}$ encrypted with their own key $E_{SK_{\mathcal{A}}}(N_{\mathcal{A}})$. However, this yields no meaningful result, as the probability of $SK_{\mathcal{A}}$ matching the actual $SK$ is negligible. Consequently, any tampering would be detected in the next comparison step $(C.I)$ or $(C.II)$.

*Impersonation Attack:* $\mathcal{A}$ attempts to impersonate a device using fabricated data. For example, to impersonate $D1$, $\mathcal{A}$ can reuse $ID_{D2}$ intercepted via eavesdropping, generate a nonce $N_{\mathcal{A}}$, encrypt it with $SK_{\mathcal{A}}$, and create $H_{\mathcal{A}}(ID_{D2}, N_{\mathcal{A}})$. However, without $SK$, $\mathcal{A}$ cannot perform correct encryption. As a result, $D2$ cannot decrypt the message, and authentication fails at step $(C.I)$. The same applies if $\mathcal{A}$ attempts to impersonate $D2$.

*Sybil Attack:* $\mathcal{A}$ forges a device identity to bypass the authentication. Similar to impersonation, this may involve reusing intercepted $IDs$ and fabricating other data, but it fails due to the absence of a valid $SK$. $\mathcal{A}$ may also attempt to flood $D2$ with requests, overloading the system and potentially causing unintended information leaks. However, our protocol prevents this by avoiding persistent storage or transmission of sensitive data and minimising its processing time.

*Brute-Force and Forging Attacks:* Guessing or forging protocol items can facilitate MitM or impersonation attacks. However, the ability to generate sufficiently secure keys enabled by an ID-PUF, combined with the use of strong cryptographic primitives (see Section 5.1), prevents $\mathcal{A}$ from successfully performing such actions, and thus from successful subsequent attacks.

*Side-Channel Attack:* $\mathcal{A}$ analyses information unintentionally leaked during protocol operation, e.g., power consumption. In such cases, security depends on the implementation of the underlying mechanisms and algorithms. According to the system model, ID-PUMA includes only strong security mechanisms; therefore, we believe it remains secure against such attacks under the defined conditions.

*Cloning Attack:* $\mathcal{A}$ creates an exact copy or model of a device, e.g., using Machine Learning (ML), to mimic its identity and behaviour. It is assumed that $\mathcal{A}$ cannot access the device's SRAM but can obtain $CC$ and the key generation algorithm. However, due to the unclonability that ID-PUF inherits from PUF, the resulting $CR_{\mathcal{A}}$ and $SK_{\mathcal{A}}$ would differ from the original, making cloning attempts ineffective.

Our security analysis shows that the proposed ID-PUMA protocol is resistant to nine known attacks. Its robustness relies on strong cryptographic primitives, freshly generated nonces, the inherent unclonability of ID-PUFs, and the ID-PUF-enabled elimination of permanent storage or transmission of the secret key.

## 6 Related Work

Most existing PUF-based authentication protocols follow a standard approach to PUF usage [17,22,26,34,35,43,46], relying on challenge-response pairs from a single PUF. Therefore, in this section, we focus instead on approaches that employ alternative techniques, which we consider more relevant to our research.

Paral *et al.* [33] introduce a reverse paradigm for PUFs where responses are public and their starting indices, serving as challenges, are secret. This scheme resists modelling attacks due to limited $CRP$ exposure and is lightweight, requiring only basic hardware components.

A lightweight mutual authentication protocol using a challenge-challenge approach is proposed in [20], where both parties share access to the same PUF. The prover sends challenges $c_{i1}$ and $c_{i2}$ such that $r_{i1} \oplus r_{i2} = r_i$, enabling for the verifier (with $c_i$) validation via XOR. The protocol supports unlimited mutual authentications and resists guessing, challenge collection and correlation attacks.

Wang *et al.* [41] present an authentication protocol based on a modified arbiter PUF. During enrolment, the server stores only a configuration signal $L$ received from the device. In the authentication phase, both parties generate response sequences using the exchanged nonces and $L$: the device via its PUF, the server via the PUF model. Authentication is performed by matching the indices of the response sequences. The protocol resists common ML attacks, including deep neural networks and logistic regression.

Majzoobi *et al.* [27] follow a similar matching approach in their Slender PUF authentication protocol. However, they use a standard arbiter PUF without modifications, relying on static PUF responses, in contrast to the dynamic approach of Wang *et al.* [41]. Despite this, the Slender protocol remains resilient to different ML attacks while maintaining low overhead.

Barbareschi *et al.* [6,5] introduce a Physical Hardware-Enabled Mutual Authentication Protocol (PHEMAP). By representing $CRP$ of recursively invoking a device's PUF as a graph, the authors use a simple path to form a chain of values, ensuring synchronisation between communicating parties throughout the authentication process. Evaluation results show that PHEMAP is resistant to MitM and differential power analysis attacks.

Xu *et al.* [42] propose a PUF matching security platform based on Programmable Delay Lines (PDL), to provide a primitive that enables low-energy and low-latency execution of multi-party communication, encryption/decryption, and authentication protocols. Using a four-step process that applies PDL to tune and modify the delays of each segment in a standard Arbiter PUF, the authors identify PUFs with identical $CRP$ mapping functions. This allows multiple devices with such PUFs to communicate securely without prior key exchange.

Chatterjee *et al.* [10] propose a Physical Related Function (PReF), an abstraction of a strong PUF, based on fully correlated $CRPs$ on two devices (i.e., $HD_{inter} \to 0$) that are typically excluded in traditional PUF use. They introduce a PReF-based key-exchange protocol that outperforms many existing PUF-based schemes in storage, computation, communication, and hardware efficiency.

This is a Pre-print: The final version will appear in the proceedings of Computer Security – ESORICS Workshops 2025 by Springer Nature.

Table 3: Comparison of our protocol with existing solutions.

| Protocol | Characteristics | | | | Attacks | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PT | CT | PK | L | EA | RA | MM | IA | SA | BF | FA | SC | CA |
| Protocols based on non-trivial PUF utilisation techniques | | | | | | | | | | | | | |
| Barbareschi *et al.* [6] | S | – | – | – | – | ✓ | ✓ | ✓ | – | – | ✓ | ✓ | – |
| Chatterjee *et al.* [10] | S | A/S | N | ✓ | – | ✓ | – | ✓ | – | – | – | ✓ | ✓ |
| Idriss *et al.* [20] | S | – | – | ✓ | – | – | – | – | – | ✓ | ✓ | – | – |
| Majzoobi *et al.* [27] | S | – | – | ✓ | ✓ | ✓ | – | – | – | ✓ | – | – | ✓ |
| Paral *et al.* [33] | S | A | – | ✓ | – | – | – | – | – | – | – | – | ✓ |
| Wang *et al.* [41] | S | – | – | – | – | – | – | – | – | ✓ | – | – | ✓ |
| Xu *et al.* [42] | S | S | N | ✓ | – | – | – | – | – | – | – | – | – |
| Protocols based on standard PUF utilisation techniques | | | | | | | | | | | | | |
| Garg *et al.* [17] | – | A | Y | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| Khatua *et al.* [22] | S | S | Y | ✓ | – | ✓ | ✓ | – | – | – | – | – | – |
| Mahalat *et al.* [26] | S/W | – | Y | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | ✓ | ✓ |
| Pham *et al.* [34] | S | S | Y | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | ✓ | – | – |
| Raj *et al.* [35] | S | S | Y | ✓ | – | ✓ | ✓ | – | – | – | – | – | – |
| Yoon *et al.* [43] | W | S | – | ✓ | – | ✓ | ✓ | ✓ | – | ✓ | – | – | ✓ |
| Zhu *et al.* [46] | W | – | Y | ✓ | – | ✓ | ✓ | ✓ | ✓ | – | ✓ | – | ✓ |
| **ID-PUMA** | S/W | S | N | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

*Characteristics:* PT: Type of used PUF (S-strong, W-weak); CT: Type of involved cryptographic mechanism (A-asymmetric, S-symmetric); PK: The need for pre-shared keys (Y-yes, N-no); L: Lightweightness of the protocol.
*Attack Abbreviations:* EA: Eavesdropping; RA: Replay; MM: Man-in-the-Middle; IA: Impersonation; SA: Sybil; BF: Brute-Force; FA: Forging; SC: Side-Channel; CA: Cloning.
*Symbols:* ✓: The protocol is resistant to the attack; –: The protocol is vulnerable to the attack or it is not mentioned in the paper.

The overview of related work shows that alternative approaches to PUF usage differ fundamentally from the ID-PUF concept, resulting in notable differences between our ID-PUMA protocol and existing ones. To highlight these distinctions, we compare our protocol with other solutions based on both standard and non-traditional PUF utilisation techniques across four key characteristics and resistance to nine attack types, as shown in Table 3.

Analysis shows that most existing protocols rely on large numbers of $CRPs$, thus requiring "strong" PUFs [6,20], while only a few support "weak" PUFs [26,43,46]. In contrast, our protocol can seamlessly use both types, demonstrating flexibility and a broader application scope; however, this remains a topic for future research. In this work, we present only a pilot version of the ID-PUF-based protocol using SRAM PUF, which is typically present in devices and does not require any enhancement or adjustments [41,42]. Additionally, unlike other protocols, ID-PUF enables the application of lightweight symmetric cryptography while eliminating the need for prior key exchange or transmission of secrets between protocol parties [34,35]. Although this paper does not detail the proof-of-concept prototype of the ID-PUMA protocol, as the main focus is on the novelty of the ID-PUF mechanism, our implementation and statistical analysis of ID-PUFs (see Sections 3.2 and 3.3) show that ID-PUF identification is feasible even on resource-constrained devices. This indicates that our ID-PUF-based protocol can be considered lightweight. Moreover, the results of the performed security analysis (see Section 5) demonstrate that ID-PUMA can resist nine known attacks, outperforming comparable protocols.

## 7 Challenges

Despite the advantages of using ID-PUFs, certain challenges may arise that can impact the real-world deployment of our ID-PUMA protocol.

*PUF Selection:* Several factors should be considered during the selection of a PUF type for an ID-PUF-based protocol. First, key generation mechanisms vary depending on PUF type, e.g., SRAM PUFs generate keys only at memory power-up, which may influence protocol design. Second, some PUFs, such as DRAM PUFs, are relatively slow, which may affect performance. Finally, not all PUFs provide the stability needed for reliable key generation.

*Scalability:* Initially, devices are paired ($D1$ and $D2$), each containing logic for its partner. The setup can scale by adding new partners, e.g., $D3$ to $D2$, requiring $D2$ to support multiple logics and increasing resource usage. Alternatively, all three devices can be mutually associated, requiring only storage of additional $IDs$ and $CCs$. Thus, the number of associated devices is scalable, though constrained by the computational and memory resources of the devices involved.

*Different Manufacturers of the Devices:* If the devices originate from different manufacturers, ID-PUF identification would need to be performed by a user-side application. This introduces development challenges due to the variety of devices and security risks if the user's system is compromised. Therefore, additional security measures are required on the user side.

## 8 Conclusions and Future Work

In this paper, we propose a novel PUF paradigm called Inter-Device PUF (ID-PUF), which leverages matching segments between two PUFs to form a shared and unique per-device-pair PUF. An ID-PUF enables devices to independently generate a shared key on demand, thereby facilitating the use of lightweight symmetric cryptography without requiring key transfer, distribution or storage. Our statistical analysis and prototype implementation confirm the feasibility of ID-PUFs on memory-constrained devices and the capability to generate a large number of unique keys. Additionally, to demonstrate the applicability of ID-PUFs, we introduce an ID-PUF-based mutual authentication protocol named ID-PUMA. Our analysis shows that it is conceptually distinct, lightweight, and offers robust security against nine known attacks, outperforming existing solutions.

Promising directions for future work include evaluating the protocol's robustness against a broader range of attacks, as well as assessing its practical feasibility across diverse device and PUF combinations.

**Acknowledgments.** This work has been funded by the Bavarian State Ministry of Science and Arts (BayStMWK), under Project "ForDaySec: Security in Everyday Use of Digital Technologies (fordaysec.de)" of the Bavarian Research Association.

**Disclosure of Interests.** The authors have no competing interests to declare that are relevant to the content of this article.

## References


1. Al-Meer, A., Al-Kuwari, S.: Physical Unclonable Functions (PUF) for IoT Devices. ACM Comput. Surv. **55**(14s) (Jul 2023). https://doi.org/10.1145/3591464
2. Andrea, I., Chrysostomou, C., Hadjichristofi, G.: Internet of Things: Security vulnerabilities and challenges. In: 2015 IEEE Symposium on Computers and Communication (ISCC). pp. 180–187 (2015). https://doi.org/10.1109/ISCC.2015.7405513
3. Arduino: Nano 33 BLE Rev2 Documentation. Arduino Documentation, https://docs.arduino.cc/hardware/nano-33-ble-rev2/
4. Arfken, G.B., Weber, H.J.: Mathematical methods for physicists. Academic Press Orlando, FL (1972)
5. Barbareschi, M., De Benedictis, A., La Montagna, E., Mazzeo, A., Mazzocca, N.: A PUF-based mutual authentication scheme for Cloud-Edges IoT systems. Future Generation Computer Systems **101**, 246–261 (2019). https://doi.org/10.1016/j.future.2019.06.012
6. Barbareschi, M., De Benedictis, A., Mazzocca, N.: A PUF-based hardware mutual authentication protocol. Journal of Parallel and Distributed Computing **119**, 107–120 (2018). https://doi.org/10.1016/j.jpdc.2018.04.007
7. Barker, E.B., Roginsky, A.L.: Transitioning the Use of Cryptographic Algorithms and Key Lengths. Tech. Rep. NIST SP 800-131A Rev. 3, National Institute of Standards and Technology (NIST), Gaithersburg, MD (2024). https://doi.org/10.6028/NIST.SP.800-131Ar3.ipd
8. Böhm, C., Hofer, M., Pribyl, W.: A microcontroller SRAM-PUF. In: 2011 5th International Conference on Network and System Security. pp. 269–273 (2011). https://doi.org/10.1109/ICNSS.2011.6060013
9. Borovkov, A.: Probability Theory. Taylor & Francis (1999)
10. Chatterjee, D., Boyapally, H., Patranabis, S., Chatterjee, U., Hazra, A., Mukhopadhyay, D.: Physically Related Functions: Exploiting Related Inputs of PUFs for Authenticated-Key Exchange. IEEE Transactions on Information Forensics and Security **17**, 3847–3862 (2022). https://doi.org/10.1109/TIFS.2022.3214089
11. CVE Details: Number of common IT security vulnerabilities and exposures (CVEs) worldwide from 2009 to 2024 YTD. Graph (08 2024), https://www.statista.com/statistics/500755/worldwide-common-vulnerabilities-and-exposures/
12. Dolev, D., Yao, A.: On the security of public key protocols. IEEE Transactions on Information Theory **29**(2), 198–208 (1983). https://doi.org/10.1109/TIT.1983.1056650
13. Durrett, R.: Probability: Theory and Examples, vol. 49. Cambridge University Press (2019)
14. Espressif Systems: ESP32-S3-DevKitC-1 Documentation. Espressif Documentation, https://docs.espressif.com/projects/esp-dev-kits/en/latest/esp32s3/esp32-s3-devkitc-1/index.html
15. Espressif Systems: ESP8266-DevKitC Getting Started Guide. PDF Documentation, https://www.espressif.com/sites/default/files/documentation/ESP8266-DevKitC_getting_started_guide__EN.pdf
16. Federal Office for Information Security (BSI): BSI Technical Guideline TR-02102-1: Cryptographic Mechanisms and Key Lengths. Technical Guideline (2025), https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-1.pdf?__blob=publicationFile&v=10
17. Garg, S., Kaur, K., Kaddoum, G., Choo, K.K.R.: Toward Secure and Provable Authentication for Internet of Things: Realizing Industry 4.0. IEEE Internet of Things Journal **7**(5), 4598–4606 (2020). https://doi.org/10.1109/JIOT.2019.2942271

18. Grubel, B.C., Bosworth, B.T., Kossey, M.R., Sun, H., Cooper, A.B., Foster, M.A., Foster, A.C.: Silicon photonic physical unclonable function. Opt. Express **25**(11), 12710–12721 (05 2017). https://doi.org/10.1364/OE.25.012710
19. Guajardo, J., Kumar, S.S., Schrijen, G.J., Tuyls, P.: FPGA Intrinsic PUFs and Their Use for IP Protection. In: Cryptographic Hardware and Embedded Systems - CHES 2007. pp. 63–80. Berlin, Heidelberg (2007)
20. Idriss, T., Bayoumi, M.: Lightweight highly secure PUF protocol for mutual authentication and secret message exchange. In: 2017 IEEE International Conference on RFID Technology & Application (RFID-TA). pp. 214–219 (2017). https://doi.org/10.1109/RFID-TA.2017.8098893
21. Katzenbeisser, S., Schaller, A.: Physical Unclonable Functions: Sicherheitseigenschaften und Anwendungen. Datenschutz und Datensicherheit - DuD **36**(12), 881–885 (Nov 2012). https://doi.org/10.1007/s11623-012-0295-z
22. Khatua, M., Das, S., Rana, P.M.: PUF-based Lightweight Mutual Authentication Scheme for IoT-based Smart Grids. In: 2024 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS). pp. 1–6 (2024). https://doi.org/10.1109/ANTS63515.2024.10898763
23. Kumar, A., Saha, R., Conti, M., Kumar, G., Buchanan, W.J., Kim, T.H.: A comprehensive survey of authentication methods in Internet-of-Things and its conjunctions. Journal of Network and Computer Applications **204**, 103414 (2022). https://doi.org/10.1016/j.jnca.2022.103414
24. Kumar, R., Burleson, W.: On design of a highly secure PUF based on non-linear current mirrors. In: 2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST). pp. 38–43 (2014). https://doi.org/10.1109/HST.2014.6855565
25. Lin, J., Yu, W., Zhang, N., Yang, X., Zhang, H., Zhao, W.: A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications. IEEE Internet of Things Journal **4**(5), 1125–1142 (2017). https://doi.org/10.1109/JIOT.2017.2683200
26. Mahalat, M.H., Karmakar, D., Mondal, A., Sen, B.: PUF-based Secure and Lightweight Authentication and Key-Sharing Scheme for Wireless Sensor Network. J. Emerg. Technol. Comput. Syst. **18**(1) (Sep 2021). https://doi.org/10.1145/3466682
27. Majzoobi, M., Rostami, M., Koushanfar, F., Wallach, D.S., Devadas, S.: Slender PUF Protocol: A Lightweight, Robust, and Secure Authentication by Substring Matching. In: 2012 IEEE Symposium on Security and Privacy Workshops. pp. 33–44 (2012). https://doi.org/10.1109/SPW.2012.30
28. Mall, P., Amin, R., Das, A.K., Leung, M.T., Choo, K.K.R.: PUF-Based Authentication and Key Agreement Protocols for IoT, WSNs, and Smart Grids: A Comprehensive Survey. IEEE Internet of Things Journal **9**(11), 8205–8228 (2022). https://doi.org/10.1109/JIOT.2022.3142084
29. Microchip Technology Inc.: 23A512 - 512 Kbit SPI Serial SRAM (2025), https://www.microchip.com/en-us/product/23a512
30. Miorandi, D., Sicari, S., De Pellegrini, F., Chlamtac, I.: Internet of Things: Vision, applications and research challenges. Ad Hoc Networks **10**(7), 1497–1516 (2012). https://doi.org/10.1016/j.adhoc.2012.02.016
31. Mohamad Noor, M.B., Hassan, W.H.: Current research on Internet of Things (IoT) security: A survey. Computer Networks **148**, 283–294 (2019). https://doi.org/10.1016/j.comnet.2018.11.025

32. National Institute of Standards and Technology (NIST): NIST Special Publication 800-78-5: Cryptographic Algorithms and Key Sizes for PIV. NIST Special Publication (2025), https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-78-5.pdf
33. Paral, Z., Devadas, S.: Reliable and efficient PUF-based key generation using pattern matching. In: 2011 IEEE International Symposium on Hardware-Oriented Security and Trust. pp. 128–133 (2011). https://doi.org/10.1109/HST.2011.5955010
34. Pham, H.L., Tran-Thi, T.Q., Bui, D.H., Tran, X.T.: Novel PUF-Based Authentication Protocol for IoT Devices with Secure Boot and Fuzzy Matching. In: 2023 International Conference on Advanced Technologies for Communications (ATC). pp. 78–83 (2023). https://doi.org/10.1109/ATC58710.2023.10318908
35. Raj, K., Bodapati, S., Chattopadhyay, A.: PUF-based Lightweight Mutual Authentication Protocol for Internet of Things (IoT) Devices. In: 2024 IEEE International Symposium on Circuits and Systems (ISCAS). pp. 1–5 (2024). https://doi.org/10.1109/ISCAS58744.2024.10558672
36. Rosen, K.H., Krithivasan, K.: Discrete mathematics and its applications, vol. 6. McGraw-Hill New York (1999)
37. Ross, S.: A first course in probability, vol. 2. Macmillan New York (1976)
38. Rührmair, U., Busch, H., Katzenbeisser, S.: Strong PUFs: Models, Constructions, and Security Proofs, p. 79–96. Springer Berlin Heidelberg (2010). https://doi.org/10.1007/978-3-642-14452-3_4
39. Samonas, S., Coss, D.: The CIA strikes back: Redefining confidentiality, integrity and availability in security. Journal of Information System Security **10**(3) (2014)
40. Statista: Internet of Things - Market Outlook Report. Statista, https://de.statista.com/statistik/studie/id/109209/dokument/internet-der-dinge-market-outlook-report/
41. Wang, Y., Wang, C., Gu, C., Cui, Y., O'Neill, M., Liu, W.: A Dynamically Configurable PUF and Dynamic Matching Authentication Protocol. IEEE Transactions on Emerging Topics in Computing **10**(2), 1091–1104 (2022). https://doi.org/10.1109/TETC.2021.3072421
42. Xu, T., Gu, H., Potkonjak, M.: An ultra-low energy PUF matching security platform using programmable delay lines. In: 2016 11th International Symposium on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC). pp. 1–8 (2016). https://doi.org/10.1109/ReCoSoC.2016.7533899
43. Yoon, S., Han, S., Hwang, E.: Joint Heterogeneous PUF-Based Security-Enhanced IoT Authentication. IEEE Internet of Things Journal **10**(20), 18082–18096 (2023). https://doi.org/10.1109/JIOT.2023.3279847
44. Zhang, Y., Ge, Y.: Evaluation of Microcontroller-Based SRAM PUF and the Authentication Scheme. In: Proceedings of the 4th International Conference on Computer, Internet of Things and Control Engineering. p. 79–86. CITCE '24, Association for Computing Machinery, New York, NY, USA (2025). https://doi.org/10.1145/3705677.3705691
45. Zhang, Z.K., Cho, M.C.Y., Wang, C.W., Hsu, C.W., Chen, C.K., Shieh, S.: IoT Security: Ongoing Challenges and Research Opportunities. In: 2014 IEEE 7th International Conference on Service-Oriented Computing and Applications. pp. 230–234 (2014). https://doi.org/10.1109/SOCA.2014.58
46. Zhu, Y., Guan, Z., Wang, Z., Li, D., Wang, Y., Xu, M.: SRAM-PUF Based Lightweight Mutual Authentication Scheme for IoT. In: 2021 IEEE Intl Conf on Parallel ISPA/BDCloud/SocialCom/SustainCom. pp. 806–813 (2021). https://doi.org/10.1109/ISPA-BDCloud-SocialCom-SustainCom52081.2021.00115