On the adoption of PUF for key agreement scheme in Internet of Things

Mario Barbareschi*
mario.barbareschi@unina.it
University of Naples "Federico II" - Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione
Naples, Italy

Valentina Casola*
casolav@unina.it
University of Naples "Federico II" - Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione
Naples, Italy

Antonio Emmanuele*
antonio.emmanuele@unina.it
University of Naples "Federico II" - Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione
Naples, Italy

Daniele Lombardi*
daniele.lombardi4@unina.it
University of Naples "Federico II" - Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione
Naples, Italy

ABSTRACT

With the rapid proliferation of Internet of Things systems, ensuring secure communication for those applications that need to exchange sensitive and/or critical data is one of the major issues to be faced. Traditional security mechanisms are often impractical due to the constrained resources typically available on IoT devices. On the other hand, Physical Unclonable Functions are emerging as one of the most promising technologies to address security-related challenges. In this manuscript, we propose a novel scheme leveraging PUF-chains to facilitate key agreement between two devices. The scheme employs a trusted third party for secure communications; additionally, it facilitates seamless and continuous modification of the cryptographic key employed, by resulting really suitable in systems for moving target defense. To demonstrate the feasibility of our proposal, we take into account an implementation of the solution on resource-constrained devices, specifically ESP8266, and conducted a thorough analysis in terms of communication and computational costs, time orhead and formal security verification.

CCS CONCEPTS

• **Security and privacy → Tamper-proof and tamper-resistant designs**; *Embedded systems security*; **Security protocols**.

KEYWORDS

Physical Unclonable Functions; Key Agreement; Internet of Things

ACM Reference Format:
Mario Barbareschi, Valentina Casola, Antonio Emmanuele, and Daniele Lombardi. 2024. On the adoption of PUF for key agreement scheme in Internet of Things. In *Proceedings of the 21st ACM International Conference on Computing Frontiers Workshops and Special Sessions (CF '24 Companion), May 7–9, 2024, Ischia, Italy.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3637543.3654656

*Authors contributed equally to this research.

1 INTRODUCTION

Nowadays, technological advances are driving down the price of sensors and microcontrollers while making it easier to connect such devices to the Internet. Combined with the growing demand for smart-things, this phenomenon is leading to the increasing diffusion of the Internet of Things (IoT). Such a paradigm is widely used in many fields ranging from healthcare [25] to smart-cities[19], bringing an increasing welfare for human society. However, despite the aforementioned advantages, the widespread use of IoT devices in daily life raises concerns about their security. Specifically, security breaches of IoT nodes, due to their pervasiveness in human life, may not only violate the privacy of their users but also harm their health [28].

Among all the security properties, mutual authentication and confidentiality are considered fundamental to guarantee in an IoT system [18]. The former ensures that only two authentic nodes are capable to correctly identify each other preventing malicious devices to be identified as authentic; on the other hand, the latter guarantees that only trusted nodes can access to the data being exchanged avoiding malicious nodes to obtain confidential information. Despite the need to respect these properties, in order to meet the high security requirements of IoT systems, classical security solutions can not be directly applied to IoT nodes due to the presence of resource constrained devices, such as Radio Frequency Identification (RFID) [16]. In fact, these devices face limitations in memory, computational capabilities, power supply when battery-packed, or silicon area, making it challenging to use traditional symmetric or asymmetric key encryption schemes. As a result, researchers have focused on developing lightweight authenticated key distribution algorithms that meet the aforementioned properties while remaining executable on such devices [13]. Furthermore these solutions, differently from the ones used in classical computing systems, must be resilient against physical attacks in which an attacker tampers directly with nodes [7]. To meet these stringent requirements, researchers are increasingly proposing Physically Unclonable Function (PUF)-based authentication and key distribution algorithms. Silicon PUFs are lightweight hardware cryptographic primitives that can be employed to generate cryptographic keys by exploiting the random fabrication imperfections of the circuit itself. For this reason, it is theoretically computationally infeasible for an attacker to be able to reproduce the behavior of a PUF, which, therefore, enjoys the property of nonclonability.

In this paper, we propose a PUF-based authenticated key-agreement scheme allowing an IoT node to mutually authenticate with a central controller by simply interrogating its PUF, without any additional cryptographic-material to be stored, hence removing the need of having secure-memories. Our solution allows for confidential data exchange between nodes without any explicit key agreement procedure, since they use as a key an already shared secret among involved entities. Moreover, since the cryptographic keys are automatically updated after each data transmission, our proposal is particularly suitable for the Moving Target Defense (MTD) mechanism, in which cryptographic material is frequently changed, making it impossible for an attacker to detect the cryptographic key used by exchanged messages [9]. We prove the security of our proposal by using the Scyther Tool [10]. Furthermore, to demonstrate the suitability of the proposal for a real IoT scenario, we implemented the protocol using the commercial micro-controller ESP8266, showing that our solution is applicable at the cost of negligible computational overhead.

The remainder of this work is structured as follows: Section 2 provides some basic preliminaries; Section 3 introduces the proposed protocol; Section 4 contains a comprehensive analysis of the proposal; Section 5 reviews the existing PUF-based mutual authentication and key exchange protocols in literature; and, finally, in Section 6, we draw the conclusion of the manuscript.

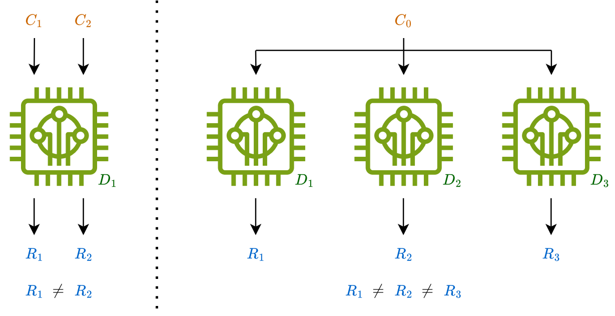


Figure 1: Typical PUF behaviour.

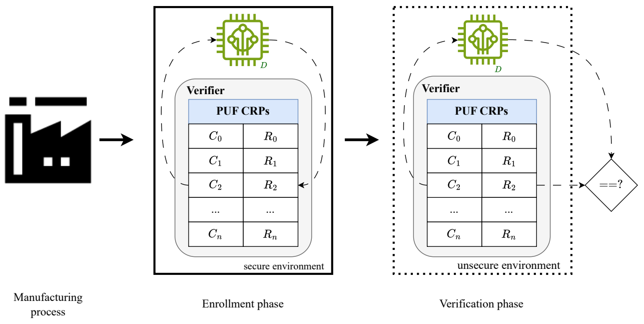


Figure 2: Typical use of PUF in security protocols, based on CRP exchange.

2 PHYSICAL UNCLONABLE FUNCTIONS

In this Section, we provide a brief overview of PUFs and the concept of PUF-chain based authentication schemes that we exploit to design our proposal.

2.1 PUFs and authentication schemes

PUFs are non-invertible functions built on the basis of nanoscale phenomena of physical objects, as for example integrated circuits, lenses, solar cells, crystals and magnets. As these phenomena are random in nature, the associated PUF is non-predictable. For instance, silicon PUFs are based on random imperfections introduced during manufacturing process of an integrated circuit, such as SRAM-PUF [8], Arbiter PUF [11] and Ring-Oscillator based PUF (RO-PUF) [23], just to cite a few examples.

PUF exhibits a behavior whereby an object which embeds one can be uniquely identified. As a matter of fact, when a PUF is stimulated with a random input, called challenge, it will produce an unpredictable output, called response. In addition, a PUF stimulated with different challenges will produce different responses; furthermore the same challenge, when submitted to different PUFs will produce different responses, as depicted in Figure 1.

The set of all possible challenge-response pairs that could be extracted from a PUF is commonly called the Challenge-Response Pair (CRP)-set. Depending on the size of that set, a PUF can be classified into strong or weak PUFs [2]. In particular, strong PUFs possess an exponentially large set of CRPs over the number of challenge bits while weak PUFs have a limited number of CRPs. For this reason, the former type is used for complex cryptographic protocols while the latter is used for key-storage [15].

Because it relies on random phenomena, a PUF may not be able to generate the same response when queried multiple times with the same challenge. For this reason, when a challenge is presented to a PUF for the first time, a bit string called helper-data is generated using a fuzzy extractor [2]. When the PUF is queried a second time with the same challenge, the helper-data is used to reconstruct the original response.

Many protocols in the scientific literature exploit the inherent properties of PUF [2] and simple exchanges of CRP to construct more complex mechanisms that aim to ensure security in communications, especially for resource-poor devices. Generally, the scheme behind these protocols involves the execution of two distinct phases, shown in Figure 2. Upon conclusion of the manufacturing process, a device D is subject to a first phase, called enrollment, performed strictly in a secure environment, i.e. not vulnerable to attacks by potential malicious agents. The purpose of the enrollment is, indeed, to establish a trustworthy reference between a device D, equipped with a PUF, and a verifier, generally with high computational and storage capabilities. To this aim, the verifier submits a random set of challenges to the PUF of device D, keeping their respective responses, along with their helper-data, in a secure memory. Once that phase is over, the device is deployed in a typically non-secure environment where, during a verification phase, the verifier interrogates it with a specific challenge and its helper-data, previously recorded, and checks whether the obtained response matches the one saved in its secure memory. If the verification is successful, the verifier can properly authenticate the device D. Such a basic scheme clearly provides only one-way authentication, since the device is not able to verify the authenticity of the verifier. Furthermore, assuming the devices are stateless, they are unable to distinguish a challenge provided by the verifier or reused by an attacker.

2.2 PUF-chains

In order to overcome the aforementioned problem and prevent the device having to save CRPs recorded by the verifier during the enrollment, thus requiring a large amount of non-volatile and secure memory, in [6], the authors propose for the first time the concepts of PUF-chains and sentinels.

Let D be a device equipped with a PUF and $\theta(\cdot)_D$ be the mathematical function representing the PUF of D, a PUF-chain $\gamma_{D,c_0,M}$ can be seen as M-iterative queries of the PUF, starting from a random challenge c_0, such that at each query the selected challenge matches the response obtained in the previous interrogation, named link l_i. For instance, $\theta^2(l_i)$ represents querying the PUF twice consecutively starting from the link l_i. During the enrollment phase of a device, it is essential that the extracted chains do not contain repeated links inside them; and, in addition, by selecting different starting challenges, randomly, the chains do not have links in common with each other. Moreover, given a link σ_0 belonging to the PUF-chain $\gamma_{D,c_0,M}$ and given a positive integer S, links that appear in the chain after σ_0 at multiple positions of S are called sentinels and constitute the sentinel set $\sum_{D,\sigma_0,S}$.

By leveraging these concepts, it is possible to use PUF-chains in applications where users can advance on the chains by consuming their links, without worrying about running into some previously used link, event that would jeopardize the security of the PUF-based application. In this regard, sentinels play a key role: while at least a number of consecutive links equal to S – sentinel period–, need to be known to properly execute a PUF-chains protocol, it is enough to expose no more than S-1 links of the chain. Following these principles, in [4–6], the authors propose a family of protocols for mutual authentication and confidential communication between nodes in Cloud-Edge domains. More specifically, thanks to an initialization procedure both verifier and device synchronize on a same link – e.g. l_i – of the used PUF-chain. Thus, during the verification phase, both entities are able to challenge the other by sending the next link in the chain, l_{i+1}, and verifying its corresponding response, l_{i+2}.

It should be noted that for the proper adoption of these protocols, the device must implement just a secure Q register in which to save the last-exchanged chain-link, limiting the required secure memory on device side to the width of a PUF response, i.e. the width of a single link. On the other hand, the verifier, differently from the device, needs to save the entire PUF-chain. Hence, the memory cost on the verifier side, for a single PUF-chain, is equal to the number of enrolled links multiplied by the width of the single link. Moreover, both the device and the verifier must have an S-module counter to skip sentinel links of the chain. Furthermore, these protocols, are completely transparent to the strategy used to reconstruct, on the device side, the original links of a chain collected during the enrollment since they consist simply of PUF queries. Recently, the authors of [1] detected and solved two attacks in PHEMAP initialization procedure: verifier impersonation attack and device traceability attack.

3 SECURE COMMUNICATION SCHEME

Based on the concepts of PUF-chains and sentinels, we present a novel scheme for secure communication of two devices relying on a Third Trusted Party (TTP), referred to as verifier, that has good computational and storage capacity. The verifier could either be the entity itself that enrolled the devices, or, in a multi-tiered network, it could be the network node closest to them, such as a gateway, that received via a secure-communication channel their PUF-chains. The proposed scheme entails two main phases, also referred as procedures: an initialization phase, shown in Figure 3, in which a device synchronizes with the verifier on a same link of its chain; and, a communication phase, shown in Figure 4, in which two devices that want to communicate with each other, encrypt their communications using a link of their chain as encryption key, by leveraging the verifier as a trusted proxy-server.

Let us suppose that devices A and B want to communicate with one each other; that they have embedded PUFs; and, that enrollment of their PUFs has been carried out in a secure environment in such a way as to extract a set of PUF-chains enjoying the properties described in Section 2.2. It should be noted that when either the verifier or devices encounter a sentinel along the chain, they bypass that particular link and proceed forward. Therefore, if l_i is marked as a sentinel, they simply utilize l_{i+1}.

Before A and B can securely communicate, an initialization phase, sketched in Figure 3, is required so that both nodes agrees on what link of the enrolled PUF-chains they should use for communications. In particular, the verifier, owing the PUF-chains of both devices, is the initiator of this procedure, also called synchronization. First, it generates a random number n and, then, it constructs two values, v_1 and v_2, by exploiting the links in the chain of the device, e.g. device A, with which it wants to synchronize, starting from a specific link, e.g. l_0, which has not yet been used. Specifically, $v_1 = l_1 \oplus l_2 \oplus n$, while $v_2 = l_3 \oplus n$. Then, it sends the plaintext message $m_1 = \{l_0, v_1, v_2\}$ to the device, which, upon receiving the message, calculates $v_1 \oplus v_2$ and checks that it is equal to the XOR operation among the three links following the link l_0 received in cleartext, i.e., $\theta(l_0) \oplus \theta^2(l_0) \oplus \theta^3(l_0)$. If the verification succeeds, A extracts the random value generated by the verifier, $n = v_2 \oplus \theta^3(l_0)$. Then, using a randomly generated value r, it calculates the values $d_1 = \theta^4(l_0) \oplus r$, $d_2 = \theta^5(l_0) \oplus (r \wedge n)$ and responds with $m_2 = \{d_1, d_2\}$. Once m_2 is received, the verifier extracts $r = l_4 \oplus d_1$ and compares d_2 with the computed $l_5 \oplus (r \wedge n)$. If the comparison succeeds, the verifier asserts that the device has obtained its nonce n and, in order to prove to A that it was able to extract nonce r, it generates $v3 = l_6 \oplus (r \vee n)$ and sends $m_3 = \{l_0, v_3\}$. The device checks that the verifier has extracted its r value by comparing v_3 with $\theta^6(l_0) \oplus (r \vee n)$ and closes the initialization procedure saving in its Q register the last used chain value, namely l_6. From this point on, both the verifier and the device are synchronized on the l_6 link of the PUF-chain.

When both A and B devices have completed the initialization procedure and are synchronized with the verifier on the l_6 and p_6

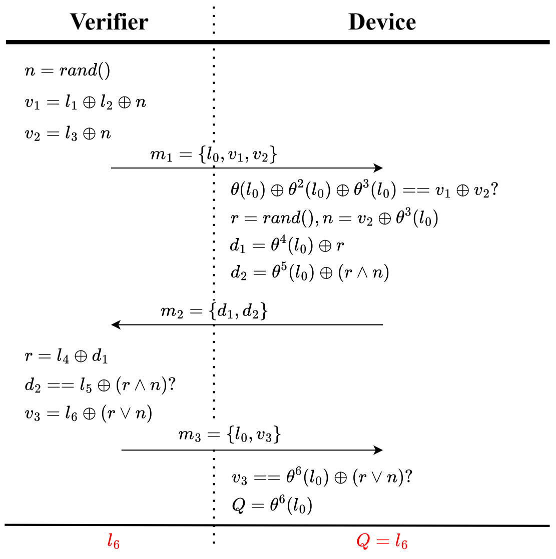

Verifier
Device
$n = rand()$
$v_1 = l_1 \oplus l_2 \oplus n$
$v_2 = l_3 \oplus n$
$m_1 = \{l_0, v_1, v_2\}$
$\theta(l_0) \oplus \theta^2(l_0) \oplus \theta^3(l_0) == v_1 \oplus v_2?$
$r = rand(), n = v_2 \oplus \theta^3(l_0)$
$d_1 = \theta^4(l_0) \oplus r$
$d_2 = \theta^5(l_0) \oplus (r \wedge n)$
$m_2 = \{d_1, d_2\}$
$r = l_4 \oplus d_1$
$d_2 == l_5 \oplus (r \wedge n)?$
$v_3 = l_6 \oplus (r \vee n)$
$m_3 = \{l_0, v_3\}$
$v_3 == \theta^6(l_0) \oplus (r \vee n)?$
$Q = \theta^6(l_0)$
l_6
$Q = l_6$

Figure 3: Initializaton phase of the proposed scheme.

links, respectively, our secure communication scheme, as illustrated in Figure 4, can be accomplished. Without loss of generality, let device A be the sender of message c to B. Bearing in mind to always skip sentinel links, device A uses the successive link to one saved in its Q register as encryption key, i.e. $\theta(Q)$; and, the successive again, i.e. $\theta^2(Q)$, as authentication tag a for the message m_4 to send. Then, it sends the message $m_4 = Enc_{\theta(Q)}(c, a)$ to the verifier. The latter deciphers the received message, by leveraging the successive link of the chain, i.e. l_7, obtaining the plaintext (c, a); then, it verifies if the received authentication tag a matches the link l_8. If this verification succeeds, the verifier can assume that the message is authentic and repeats the same procedure adopted by device A, to redirect the message c to device B. This time, however, the verifier utilizes the links of device B as encryption key and authentication tag, b, i.e. p_7 and p_8 respectively, in order to send the message $m_5 = Enc_{p_7}(c, b)$. Device B deciphers the message and verifies the authenticity of its content by consuming two links of its chain. Finally, it sends an outcome of these operations to the verifier, via message m_6, which, in turn, communicates redirects it to device A, through message m_7. Clearly, by adopting the same encryption key selection and authentication tag procedures, two more links from the respective device chains are consumed for both messages. This is the reason why, at the end of the communication procedure, if all is successful, devices A and B will be synchronized with the verifier on the l_{10} and p_{10} links, respectively. Note that in case, for whatever reason, the verifier loses synchronization with device B, there would be no m_6 message; nevertheless, the verifier could still provide the negative outcome of the communication to device A –for instance, by activating a timer–, preserving its synchronization on the l_{10} link.

Bear in mind that the scheme just described, expects to use a sentinel period of 4, however it can be changed. In that case, it is also necessary to modify the way messages m_1, m_2 and m_3 are

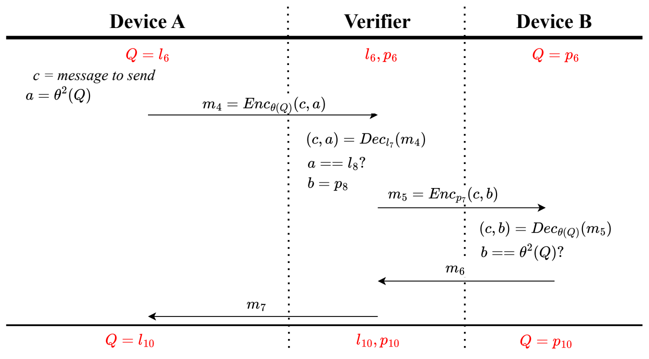


Figure 4: Communication phase of the proposed scheme.

constructed, by leveraging the equations:

$$\begin{aligned} m_1 &= \left\{ l_i, \left(\sum_{j=0}^{S-3} l_{i+1+j} \right) \oplus n, l_{i+S-1} \oplus n \right\} = \{l_i, v_1, v_2\} \\ m_2 &= \{l_{i+S} \oplus r, l_{i+S+1} \oplus (r \wedge n)\} = \{d_1, d_2\} \\ m_3 &= \{l_i, l_{i+S+2} \oplus (r \vee n)\} = \{l_i, v_3\} \end{aligned} \tag{1}$$

introduced in [6]. The protocol is resilient to transmission errors by simply starting a new synchronization procedure among the device and the verifier. Moreover, in case of packet loss, each involved entity can implement a timer and restart a new synchronization procedure after the timer is expired.

As final remark, we point out that our proposal is also suitable when a device establishes multiple end-to-end communications with different devices. If a device embeds only one secure Q register, all communications must be serialized to maintain synchronization between the verifier and the initiator – i.e., before another secure transmission begins, a device must receive the m_7 message from the verifier. However, if a device has the ability to store multiple secure registers, then – unlike the previous solution –, subsequent communication can begin before the m_7 message is received, by using parallel synchronizations on different PUF chains. Of course, this approach must be supported accordingly on the verifier side, by synchronizing with the same device on different chains.

4 PROTOCOL EVALUATION

This Section provides to readers an in depth evaluation in terms of computational costs, communication costs, time overhead and formal security of the proposed protocol.

4.1 Computational costs analysis

In order to prove the lightweight nature of the protocol, for each one of its procedures, we investigate computational costs on both device and verifier sides. To make this analysis agnostic to the specific target implementation, we consider this cost as the sum of the different operations involved, as shown in Table 1.

It is evident that both the initialization and communication phases require symmetric operations for both the node and the verifier. Moreover, the PUF query for a device corresponds to a memory access for the verifier since it checks the validity of a link by accessing its memory. As demonstrated in [6] this can introduce

Table 1: Computational cost of the protocol in each different phase. Each operation type is indicated with its name in uppercase. XOR: bitwise XOR operation. MEM: memory access. PUF: PUF query. AND: bitwise and operation. OR: bitwise or operation.

	Initialization phase	Communication phase
Device	$(\mathrm{S}+2)\cdot(\mathrm{MEM}+\mathrm{XOR})+AND+OR$	$ENC+DEC+4\cdot\mathrm{PUF}$
Verifier	$(\mathrm{S}+2)\cdot(\mathrm{PUF}+\mathrm{XOR})+AND+OR$	$ENC+DEC+4\cdot\mathrm{MEM}$

more computational overhead on the verifier side since the time required to obtain a PUF response is lower compared to a memory access.

Except from PUF queries and memory accesses, the initialization phase requires only simple binary operations, reducing the computational overhead of an initialization procedure to a minimum.

The communication phase instead relies on symmetric encryption and decryption operations. Therefore, the computational cost depends on the type of encryption algorithm used. For this reason, lightweight cryptographic algorithm should be used; in particular, Chaskey [24] results to be one of the best performing in terms of execution time and memory usage according to the FELICS benchmark [1]. As final remark, it is worth noticing that during a communication the three involved entities perform encryption and decryption operations on data of different sizes. The device starting the communication encrypts the message to cipher with a link of its PUF-chain, and deciphers the content of message m_7 corresponding to the size of a link of a PUF-chain. Similarly, the receiver decrypts the received ciphertext and encrypts the PUF-chain link used to complete the procedure on its side.

4.2 Communication costs analysis

One of the key requirements of authentication and key exchange protocols is having minimum impact on the network overhead. For this reason, similarly to the computational analysis, we calculate the communication overhead of our proposal in each one of its phases. Specifically, by indicating with Size(.) the size of a specific value, we consider the total overhead as the sum of the sizes of the exchanged messages.

Given that all l_i links in a PUF-chain have the same size, the initialization phase causes a network overhead of $7\cdot\mathrm{Size}(l_i)$. This constant value indicates that the initialization has minimum impact on the overall network traffic; for example, by assuming to use PUF responses of 256 bits, the total overhead would be 1792 bits.

Conversely, the communication phase depends on the size of the ciphertext. Specifically, message m_4 and m_5, by supposing to use a block cipher, requires $2\cdot\mathrm{Size}(c)+2\cdot\mathrm{Size}(l_i)$ in which c indicates the ciphertext. The last two messages of this phase introduce a constant overhead of $2\cdot\mathrm{Size}(l_i)$. Since the communication overhead of a classical ciphertext transmission is $\mathrm{Size}(c)$ the total introduced overhead of the communication phase, considering all the message, is $\mathrm{Size}(c)+4\cdot\mathrm{Size}(l_i)$. Therefore, in light of the fact that the introduced overhead increases linearly with the size of the c, we can infer that the protocol is scalable w.r.t. the size of the ciphertext.

[1]https://www.cryptolux.org/index.php/FELICS

4.3 Formal security analysis

Formally verifying the security of communication protocols typically involves the utilization of specialized analysis tools such as Vispa, ProVerif, Scyther, and Spin. Through these tools, it is possible to implement a model describing the behavior of the system under analysis using their specific language. Each entity involved in the protocol, such as verifier and device, is assigned to a specific role, detailing the primary actions it performs within the system. For our study, in particular, we utilize Scyther [10], one of the most widely adopted security tools in the scientific community, which leverages the Security Protocol Description Language (SPDL). We adopt the Dolev-Yao model [12], where an attacker can eavesdrop on all packets on the network and impersonate any device. The only limitation to his capabilities are cryptographic operations such as encryption. The Figure 5 presents the obtained results, while Listing 1 shows the SPDL model used to verify the initialization procedure of the proposed scheme.

Listing 1: SPDL model for the initialization procedure of the proposed scheme.

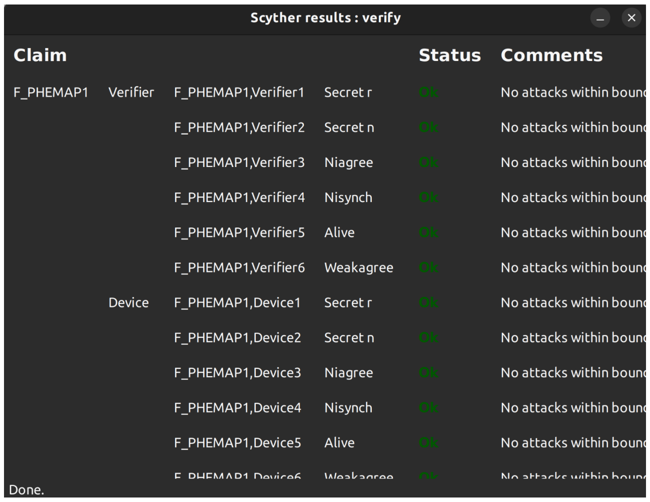
```
const PUF: Function;
const XOR: Function;
const AND: Function;
const OR: Function;

macro l1 = PUF(l0);
macro l2 = PUF(l1);
macro l3 = PUF(l2);
macro l4 = PUF(l3);
macro l5 = PUF(l4);
macro l6 = PUF(l5);

macro v1 = XOR(XOR(l1, l2),n);
macro v2 = XOR(l3,n);
macro d1 = XOR(l4,r);
macro d2 = XOR(l5,AND(r,n));
macro v3 = XOR(l6,OR(r,n));

protocol F-PHEMAP1(Verifier,Device)
{
role Verifier
{
secret l0;
fresh r,n: Nonce;
send_1 (Verifier,Device,l0,v1,v2);
recv_2 (Device,Verifier,d1,d2);
match (d2,XOR(l5,AND(r,n)));
send_3 (Verifier,Device,v3);
claim (Verifier, Secret, r);
claim (Verifier, Secret, n);
claim (Verifier,Niagree); claim (Verifier,Nisynch);
claim (Verifier,Alive); claim (Verifier,Weakagree);
};

role Device
```



```
Scyther results : verify

Claim                                                    Status  Comments

F_PHEMAP1  Verifier  F_PHEMAP1,Verifier1  Secret r       Ok      No attacks within boun
                     F_PHEMAP1,Verifier2  Secret n       Ok      No attacks within boun
                     F_PHEMAP1,Verifier3  Niagree        Ok      No attacks within boun
                     F_PHEMAP1,Verifier4  Nisynch        Ok      No attacks within boun
                     F_PHEMAP1,Verifier5  Alive          Ok      No attacks within boun
                     F_PHEMAP1,Verifier6  Weakagree      Ok      No attacks within boun
           Device    F_PHEMAP1,Device1    Secret r       Ok      No attacks within boun
                     F_PHEMAP1,Device2    Secret n       Ok      No attacks within boun
                     F_PHEMAP1,Device3    Niagree        Ok      No attacks within boun
                     F_PHEMAP1,Device4    Nisynch        Ok      No attacks within boun
                     F_PHEMAP1,Device5    Alive          Ok      No attacks within boun
                     F_PHEMAP1,Device6    Weakagree      Ok      No attacks within boun
Done.
```

Figure 5: Results of formal verification with Scyther tool

```
{
secret l0;
fresh r,n: Nonce;
recv_1 (Verifier,Device,l0,v1,v2);
match (XOR(v1,v2),XOR(l1,XOR(l2,l3)));
send_2 (Device,Verifier,d1,d2);
recv_3 (Verifier,Device,v3);
match (v3,XOR(l6,OR(r,n)));
claim (Device, Secret, r);
claim (Device, Secret, n);
claim (Device,Niagree); claim (Device,Nisynch);
claim (Device,Alive); claim (Device,Weakagree);
};

};
```

4.4 Time overhead analysis

To assess the lightness of the scheme and hence its suitability for resource-constrained devices, we evaluate also the overall time overhead, intended as the time required to perform the initialization and communication phases. To this purpose, we select the ESP8266 board, equipped with a NodeMCU Lolin V3, as device node; and, an Avnet Ultra96v2 with MPSoC ZynqUltrascale+ as verifier node. For what pertains the PUF, since our scheme does not make any assumption on the specific PUF, except for the adoption of strong PUF, we use an abstract PUF implemented with the BLAKE hash function of Crypto library[2].

In Figure 6, we show the timing trend of the initialization phase as the sentinel period changes, using 128-bit links. The trend of the curve suggests, as previously analyzed, that the initialization time increases linearly with respect to the sentinel period. For this reason, the choice of this parameter implies a trade-off among security and the overall overhead of the initialization phase. On the other hand, the timings of the communication phase depends mostly on the size of the ciphertext and on the specific cryptographic algorithm used. As example, using a custom implementation of Chaskey [24], with

[2]https://www.arduino.cc/reference/en/libraries/crypto

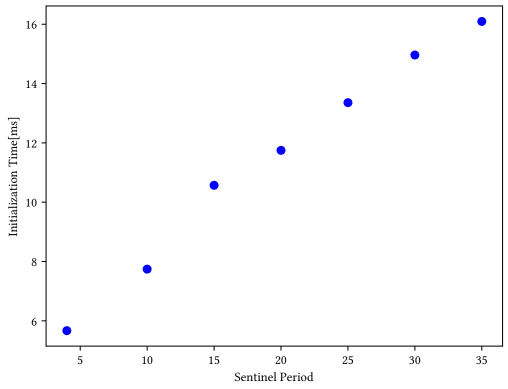


Figure 6: Time overhead as the sentinel period changes

a message of size 1024-bytes, the entire procedure is completed in an average time of 138 ms.

5 RELATED WORK

Silicon PUFs are being increasingly used to design lightweight and secure communication protocols since the time required to compute a response is low and they are hard to predict [14]. This advantage, combined with the simplicity of implementing one even in commercial microcontrollers [3], is leading to an increasing adoption of these circuits to design security countermeasures for IoT domains.

As previously mentioned, resource constrained devices benefit from the use of PUFs in order to execute lightweight cryptographic protocols. For RFIDs, in fact, the authors of [29] propose a PUF-based protocol that involves the use of simple hash functions and XOR operations to perform a mutual authentication in three exchanged messages. Another solution, designed for RFID, is the one proposed [21] that entails the use of Deep Learning (DL). In particular, the authors propose a custom enrollment phase in which, from an arbiter PUF embedded in the tag, a DL model is trained by using a large CRPs set and stored into a secure database. Therefore, the authentication procedure requires the tag reader to ask the database for the model of a specific tag, so that it can validate the responses to the challenges submitted to the arbiter PUF of the latter.

The capability of modelling a strong PUF upon collecting a large number of CRPs can also be used from an attacker to predict PUFs responses [27]. For this reason the authors of [17] propose a protocol by leveraging the concept of pseudo-challenge. The latter is a binary-string sent in clear text and modified by a device using a transformation function that maps the pseudo-challenge to another value used as challenge. This approach avoids obtaining the CRPs set by simply eavesdropping the exchanged messages making difficult for an attacker to model the underlying PUF. A concept similar to pseudo-challenges is the extended Challenge Response Pair (eCRP) [22] that enables mutual authentication between two IoT nodes without the involvement of a remote TTP. In particular, an eCRP consists on a quintuple of three different challenges and two binary-strings, obtained by combining the PUF responses of two devices using the XOR operation. The authors propose a protocol comprising two distinct phases. At first, two nodes mutually authenticate with a PUF equipped gateway node, leveraging the eCRP of the latter, collected during the enrollment. In the second phase, the devices perform an authentication and key agreement procedure with each other. A similar approach is the one proposed in [22] in which a local server mutually authenticates with a pair of nodes and distributes a key-mask pair to each one, allowing them to perform a direct key agreement procedure among each other. Another approach that minimises the involvement of a remote server is that proposed by the authors of [20]. Their protocol, by leveraging Elliptic Curve Cryptography (ECC) based Certificateless-Public Key Cryptography (CL-PKC) allows mutual authentication and key exchange between two end nodes, guaranteeing perfect forward secrecy and solving the key escrow problem-i.e. the server can not obtain the session key established between two legitimate participants.

A different defense-strategy for modelling attacks is the one proposed in [26]. In fact, the proposed protocol envisages that a device first authenticates a server, using other cryptographic primitives such as True Random Number Generator (TRNG), and only then uses its PUF. This approach prevents brute force attacks in which an attacker tries to obtains a large number CRPs by iteratively launching authentication procedures.

6 CONCLUSION

In this manuscript, we have proposed a novel scheme aimed at enabling secure end-to-end communication within an IoT system, leveraging the PUF of the involved devices. By harnessing the notion of PUF-chains, the new scheme initially establishes synchronization between a device and a verifier. Subsequently, based on this synchronization, communication between two devices can be encrypted utilizing the next link of the chain on which devices are syncronized. We conducted an in depth analysis of the costs associated with our proposal, revealing its efficiency and applicability even to devices with limited resources in terms of storage and computation. Finally, we formally verified the security of the scheme employing the verification tool Scyther.

REFERENCES

[1] Morteza Adeli, Nasour Bagheri, Honorio Martín, and Pedro Peris-Lopez. 2022. Challenging the security of "A PUF-based hardware mutual authentication protocol". *J. Parallel and Distrib. Comput.* 169 (Nov. 2022), 199–210. https://doi.org/10.1016/j.jpdc.2022.06.018

[2] Mario Barbareschi. 2017. Notions on silicon physically unclonable functions. In *Hardware security and trust: Design and deployment of integrated circuits in a threatened environment*, Nicolas Sklavos, Ricardo Chaves, Giorgio Di Natale, and Francesco Regazzoni (Eds.). Springer International Publishing, Cham, 189–209. https://doi.org/10.1007/978-3-319-44318-8_10

[3] Mario Barbareschi, Valentina Casola, Alessandra De Benedictis, Erasmo La Montagna, and Nicola Mazzocca. 2021. On the Adoption of Physically Unclonable Functions to Secure IIoT Devices. *IEEE Transactions on Industrial Informatics* 17, 11 (Nov. 2021), 7781–7790. https://doi.org/10.1109/TII.2021.3059656 Conference Name: IEEE Transactions on Industrial Informatics.

[4] Mario Barbareschi, Valentina Casola, and Daniele Lombardi. 2023. Lightweight Secure Keys Management Based on Physical Unclonable Functions. In *2023 9th International Workshop on Advances in Sensors and Interfaces (IWASI)*. IEEE, New York, NY, USA, 34–39. https://doi.org/10.1109/IWASI58316.2023.10164402 ISSN: 2836-7936.

[5] Mario Barbareschi, Alessandra De Benedictis, Erasmo La Montagna, Antonino Mazzeo, and Nicola Mazzocca. 2019. A PUF-based mutual authentication scheme for Cloud-Edges IoT systems. *Future Generation Computer Systems* 101 (Dec. 2019), 246–261. https://doi.org/10.1016/j.future.2019.06.012

[6] Mario Barbareschi, Alessandra De Benedictis, and Nicola Mazzocca. 2018. A PUF-based hardware mutual authentication protocol. *J. Parallel and Distrib. Comput.* 119 (Sept. 2018), 107–120. https://doi.org/10.1016/j.jpdc.2018.04.007

[7] Ismail Butun, Patrik Osterberg, and Houbing Song. 2020. Security of the Internet of Things: Vulnerabilities, Attacks, and Countermeasures. *IEEE Communications Surveys & Tutorials* 22, 1 (2020), 616–644. https://doi.org/10.1109/COMST.2019.2953364

[8] Christoph Böhm, Maximilian Hofer, and Wolfgang Pribyl. 2011. A microcontroller SRAM-PUF. In *2011 5th International Conference on Network and System Security*. IEEE, New York, NY, USA, 269–273. https://doi.org/10.1109/ICNSS.2011.6060013

[9] Jin-Hee Cho, Dilli P. Sharma, Hooman Alavizadeh, Seunghyun Yoon, Noam Ben-Asher, Terrence J. Moore, Dong Seong Kim, Hyuk Lim, and Frederica F. Nelson. 2020. Toward Proactive, Adaptive Defense: A Survey on Moving Target Defense. *IEEE Communications Surveys & Tutorials* 22, 1 (2020), 709–745. https://doi.org/10.1109/COMST.2019.2963791

[10] Cas J. F. Cremers. 2008. The Scyther Tool: Verification, Falsification, and Analysis of Security Protocols. In *Computer Aided Verification*, Aarti Gupta and Sharad Malik (Eds.). Springer, Berlin, Heidelberg, 414–418. https://doi.org/10.1007/978-3-540-70545-1_38

[11] Daihyun Lim, J.W. Lee, B. Gassend, G.E. Suh, M. Van Dijk, and S. Devadas. 2005. Extracting secret keys from integrated circuits. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 13, 10 (Oct. 2005), 1200–1205. https://doi.org/10.1109/TVLSI.2005.859470

[12] D. Dolev and A. Yao. 1983. On the security of public key protocols. *IEEE Transactions on Information Theory* 29, 2 (March 1983), 198–208. https://doi.org/10.1109/TIT.1983.1056650

[13] Mohammed El-hajj, Ahmad Fadlallah, Maroun Chamoun, and Ahmed Serhrouchni. 2019. A Survey of Internet of Things (IoT) Authentication Schemes. *Sensors* 19, 5 (Jan. 2019), 1141. https://doi.org/10.3390/s19051141 Number: 5 Publisher: Multidisciplinary Digital Publishing Institute.

[14] Prosanta Gope, Jemin Lee, and Tony Q. S. Quek. 2018. Lightweight and Practical Anonymous Authentication Protocol for RFID Systems Using Physically Unclonable Functions. *IEEE Transactions on Information Forensics and Security* 13, 11 (Nov. 2018), 2831–2843. https://doi.org/10.1109/TIFS.2018.2832849 Conference Name: IEEE Transactions on Information Forensics and Security.

[15] Charles Herder, Meng-Day Yu, Farinaz Koushanfar, and Srinivas Devadas. 2014. Physical Unclonable Functions and Applications: A Tutorial. *Proc. IEEE* 102, 8 (Aug. 2014), 1126–1141. https://doi.org/10.1109/JPROC.2014.2320516 Conference Name: Proceedings of the IEEE.

[16] Hung-Yu Chien. 2007. SASI: A New Ultralightweight RFID Authentication Protocol Providing Strong Authentication and Strong Integrity. *IEEE Transactions on Dependable and Secure Computing* 4, 4 (Oct. 2007), 337–340. https://doi.org/10.1109/TDSC.2007.70226

[17] Tarek A. Idriss, Haytham A. Idriss, and Magdy A. Bayoumi. 2021. A Lightweight PUF-Based Authentication Protocol Using Secret Pattern Recognition for Constrained IoT Devices. *IEEE Access* 9 (2021), 80546–80558. https://doi.org/10.1109/ACCESS.2021.3084903

[18] Waseem Iqbal, Haider Abbas, Mahmoud Daneshmand, Bilal Rauf, and Yawar Abbas Bangash. 2020. An In-Depth Analysis of IoT Security Requirements, Challenges, and Their Countermeasures via Software-Defined Security. *IEEE Internet of Things Journal* 7, 10 (Oct. 2020), 10250–10276. https://doi.org/10.1109/JIOT.2020.2997651

[19] Latif U. Khan, Ibrar Yaqoob, Nguyen H. Tran, S. M. Ahsan Kazmi, Tri Nguyen Dang, and Choong Seon Hong. 2020. Edge-Computing-Enabled Smart Cities: A Comprehensive Survey. *IEEE Internet of Things Journal* 7, 10 (Oct. 2020), 10200–10232. https://doi.org/10.1109/JIOT.2020.2987070 Conference Name: IEEE Internet of Things Journal.

[20] Sensen Li, Tikui Zhang, Bin Yu, and Kuan He. 2021. A Provably Secure and Practical PUF-Based End-to-End Mutual Authentication and Key Exchange Protocol for IoT. *IEEE Sensors Journal* 21, 4 (Feb. 2021), 5487–5501. https://doi.org/10.1109/JSEN.2020.3028872 Conference Name: IEEE Sensors Journal.

[21] Wei Liang, Songyou Xie, Dafang Zhang, Xiong Li, and Kuan-ching Li. 2021. A Mutual Security Authentication Method for RFID-PUF Circuit Based on Deep Learning. *ACM Transactions on Internet Technology* 22, 2 (2021), 34:1–34:20. https://doi.org/10.1145/3426968

[22] Karim Lounis and Mohammad Zulkernine. 2021. T2T-MAP: A PUF-Based Thing-to-Thing Mutual Authentication Protocol for IoT. *IEEE Access* 9 (2021), 137384–137405. https://doi.org/10.1109/ACCESS.2021.3117444 Conference Name: IEEE Access.

[23] Abhranil Maiti and Patrick Schaumont. 2011. Improved Ring Oscillator PUF: An FPGA-friendly Secure Primitive. *Journal of Cryptology* 24, 2 (April 2011), 375–397. https://doi.org/10.1007/s00145-010-9088-4

[24] Nicky Mouha, Bart Mennink, Anthony Van Herrewege, Dai Watanabe, Bart Preneel, and Ingrid Verbauwhede. 2014. Chaskey: An Efficient MAC Algorithm for 32-bit Microcontrollers. In *Selected Areas in Cryptography – SAC 2014*, Antoine Joux and Amr Youssef (Eds.). Springer International Publishing, Cham, 306–323. https://doi.org/10.1007/978-3-319-13051-4_19

[25] Yazdan Ahmad Qadri, Ali Nauman, Yousaf Bin Zikria, Athanasios V. Vasilakos, and Sung Won Kim. 2020. The Future of Healthcare Internet of Things: A Survey of Emerging Technologies. *IEEE Communications Surveys & Tutorials* 22, 2 (2020), 1121–1167. https://doi.org/10.1109/COMST.2020.2973314

[26] Mahmood Azhar Qureshi and Arslan Munir. 2022. PUF-RAKE: A PUF-Based Robust and Lightweight Authentication and Key Establishment Protocol. *IEEE Transactions on Dependable and Secure Computing* 19, 4 (July 2022), 2457–2475. https://doi.org/10.1109/TDSC.2021.3059454

[27] Ulrich Rührmair, Jan Sölter, Frank Sehnke, Xiaolin Xu, Ahmed Mahmoud, Vera Stoyanova, Gideon Dror, Jürgen Schmidhuber, Wayne Burleson, and Srinivas Devadas. 2013. PUF Modeling Attacks on Simulated and Silicon Data. *IEEE Transactions on Information Forensics and Security* 8, 11 (Nov. 2013), 1876–1891. https://doi.org/10.1109/TIFS.2013.2279798 Conference Name: IEEE Transactions on Information Forensics and Security.

[28] Bruce Schneier. 2018. *Click Here to Kill Everybody: Security and Survival in a Hyper-connected World*. W. W. Norton & Company, New York, NY, USA. Google-Books-ID: BGZSDwAAQBAJ.

[29] Feng Zhu, Peng Li, He Xu, and Ruchuan Wang. 2019. A Lightweight RFID Mutual Authentication Protocol with PUF. *Sensors* 19, 13 (Jan. 2019), 2957. https://doi.org/10.3390/s19132957 Number: 13 Publisher: Multidisciplinary Digital Publishing Institute.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009