INTERNATIONAL STANDARD

ISO/IEC 20897-1

First edition
2020-12

Information security, cybersecurity and privacy protection — Physically unclonable functions —

Part 1:
Security requirements

Sécurité de l'information, cybersécurité et protection de la vie privée — Fonctions non clonables physiquement —

Partie 1: Exigences de sécurité

Reference number
ISO/IEC 20897-1:2020(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see http://patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *Information security, cybersecurity and privacy protection*.

A list of all parts in the ISO/IEC 20897 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

This document specifies the security requirements for physically unclonable functions (PUFs) for generating non-stored cryptographic parameters.

Cryptographic modules generate the certain class of critical security parameters such as a secret key using a random bit generator within the modules. Such modules can store generated security parameters in embedded non-volatile memory elements. For higher security, a combination of tamper response and zeroization techniques may be used for protecting stored security parameters from active unauthorized attempts of accessing such parameters. However, as the reverse-engineering technology advances, the risk of theft of such stored security parameters has become higher than ever.

The rapidly pervading technology called a PUF is promising to mitigate the above-mentioned risks by enabling security parameter management without storing such parameters. PUFs are hardware-based functions providing steadiness and randomness of their outputs and physical and mathematical unclonability of the functions themselves, taking advantage of intrinsic subtle variations in the device's physical properties, which are also considered object's fingerprints. PUFs can be used for security parameter generation (e.g. key, initialization vector, nonce and seed), entity authentication or device identification in cryptographic modules.

Now, security requirements of PUFs should be considered at system level, meaning that they should consider many possible attack paths, as detailed further in this document.

The purpose of this document is to define the security requirements of batches of PUFs and of single instances of PUF for assuring an adequate level of quality of the provided PUFs in cryptographic modules. This document is meant to be used for the following purposes.

a) In the procurement process of a PUF-equipped product, the procurement body specifies the security requirements of the PUF in accordance with this document. The product vendor evaluates the PUF whether the PUF satisfies all the specified security requirements, and reports the evaluation results to the procurement body.

b) The vendors evaluate the security of their PUF, publicize the evaluation results and clarify the security of their PUF.

It should be noted that all of the security requirements defined in this document are not necessarily quantitatively evaluable.

This document is related to ISO/IEC 19790 which specifies security requirements for cryptographic modules. In those modules, CSPs (e.g. key) and PSPs [e.g. identifier (ID)] are the assets to protect. PUF is one solution to avoid storing security parameters, thereby increasing the overall security of a cryptographic module.

Information security, cybersecurity and privacy protection — Physically unclonable functions —

Part 1: Security requirements

1 Scope

This document specifies the security requirements for physically unclonable functions (PUFs). Specified security requirements concern the output properties, tamper-resistance and unclonability of a single and a batch of PUFs. Since it depends on the application which security requirements a PUF needs to meet, this documents also describes the typical use cases of a PUF.

Amongst PUF use cases, random number generation is out of scope in this document.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 18031, *Information technology — IT Security techniques — Random bit generation*

ISO/IEC 19790, *Information technology — Security techniques — Security requirements for cryptographic modules*

3 Terms and definitions

For the purposes of this document, terms and definitions given in ISO/IEC 18031, ISO/IEC 19790 and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1
challenge
variable parameters input to a PUF

Note 1 to entry: Some type of PUFs do not take an input challenge, and such PUFs are called a no-challenge PUF. A no-challenge PUF can be seen as a special type of PUF where a challenge length is 0 bit (see 3.9).

3.2
confined PUF
DEPRECATED: weak PUF
PUF that has a limited space of challenge-response pairs

Note 1 to entry: The term “weak PUF” does not properly express the characteristics of the PUF; nonetheless, it is the way this category of PUFs is referred to in the scientific literature.

3.3
extensive PUF
DEPRECATED: strong PUF
PUF that has so large space of challenge-response pairs that not all addresses cannot be read out within the attack time scales and its entire function cannot be modelled in extenso from the knowledge of few challenge/response pairs on a different device (e.g. a general purpose processor)

3.4
false acceptance rate
FAR
probability that the inter-distance is smaller than or equal to the set threshold

Note 1 to entry: FAR is equivalent to the evaluation of the cumulative distribution function of the inter-distance at the set threshold.

3.5
false rejection rate
FRR
probability that the intra-distance is larger than the set threshold

Note 1 to entry: FRR is equivalent to the complement of the evaluation of the cumulative distribution function of the intra-distance at the set threshold.

3.6
fuzzy extractor
scheme to recover the original data from noisy data using error-correcting technique

Note 1 to entry: Fuzzy extractor uses a pair of algorithms; the first one generates a helper string from original data, and the second one recovers the original data from noisy data using error correcting techniques and the helper string.

3.7
inter-Hamming distance
inter-HD
Hamming distance between the two responses obtained by giving the identical challenges to two different PUFs or obtained from two different no-challenge PUFs

3.8
intra-Hamming distance
intra-HD
Hamming distance between the two responses obtained by giving the same challenge twice to the same PUF or obtained from the same no-challenge PUF

3.9
no-challenge PUF
one form of a confined PUF that does not receive a challenge as an input

3.10
physically unclonable function
physical unclonable function
PUF
function implemented in a device that is produced or configured with the security objective that random fluctuations in the production process lead to different behaviours and are hard to reproduce physically

3.11
response
output value from a PUF

3.12
static entropy source
source of entropy which is prepared in advance of its use

4 Abbreviated terms

CRP	challenge response pair
CSP	critical security parameter
DFA	differential fault analysis
DRBG	deterministic random bit generator
ECC	error correction code
FIB	focused ion beam
HCI	hot carrier injection
HMAC	hash-based message authentication code
LVP	laser voltage probing
MOS	metal oxide semiconductor
NBTI	negative bias temperature instability
PSP	public security parameter
PUF	physically unclonable function

5 Security requirements for PUFs

5.1 General

A PUF is a physical object that provides a unique digital output as a (deterministic) function of the given inputs. The mapping between the inputs and outputs of a PUF is determined by the variation of device components (e.g. transistor, wire, capacitance) arisen during manufacturing process, etc. Since the device variation is random and uncontrollable, it is virtually impossible to manufacture two PUFs that behave in exactly the same way. Actually, the idea behind a PUF is that given the identical “blueprint” or fabrication file, every instance behaves differently. Therefore, the PUF may be used for security parameter generation, device identification and device authentication.

A PUF’s output is expected to be random for different challenges and for inter-modules. Another random sequence or key to be generated from a PUF should not be estimated from any other output of the PUF. In order to keep using the same generated key, the same challenge only has to be kept. Even if a challenge is leaked out, the corresponding response should not be estimated without having the same device activated. However, for those purposes, particular caution should be taken due to the PUF’s properties. The raw output of a PUF contains a small amount of errors due to subtle fluctuations in the physical properties exploited. Therefore, typically an error correction scheme is combined with the output of a PUF in order to make the output the same every time the same challenge is given. In order to avoid undesired regeneration of the same key or nonce, it is required to maintain the challenge value carefully.

A PUF and DRBG are different in the following two aspects:

— an output of a DRBG is completely computable by a deterministic algorithm given a seed, whereas a response of a PUF is virtually impossible to compute from a challenge;

— different instances of a DRBG will produce completely the same outputs given the same seed, whereas different PUFs will output different responses.

Therefore, the application of a PUF for random number generation is out of scope in this document.

A PUF is seen as an instance of a design by a factory (which can be modelled as a random variable). Ideally, the properties of each PUF instance are independent and identically distributed (i.i.d.), as represented in Figure 1. Notice that, in practice, the PUFs are neither identical nor independent: it is the purpose of some security requirements to quantify how well PUF instances differ one from each other.

For the PUF to indeed provide a securely strong and reliable response, some properties shall be met. This clause concerns the definition of the security requirements which apply irrespective of PUF implementation, whereas there exist many kinds of PUF implementations (refer to Annex A).

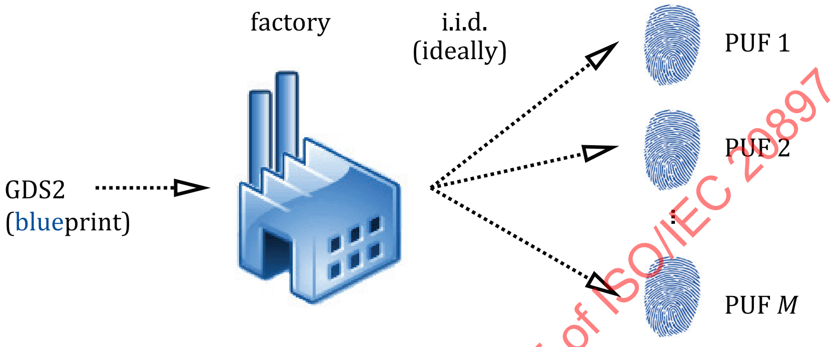

Figure 1 — (Ideal) modelization of a PUF as a random instance from a factory

5.2 PUF interface

Generally, a PUF receives an input called a *challenge*, and generates a unique output called a *response*. A challenge is a bit-string made up of *challengeLength* bits, and a response is a bit-string made up of *responseLength* bits. The response may be specific to an input challenge. In case of a no-challenge PUF, the challenge length is considered as 0 bit. Thus, a PUF can be an alternative to a memory of $2^{challengeLength}$ words where each word is *responseLength* bits.

A PUF may have optional ports, which allow for security testing. The signals from these ports are output status, and are meant to indicate whether the PUF responses can be used securely. After the security test is finished, these ports shall be disabled to avoid being exploited by an attacker.

In addition, a PUF may have other optional ports to control the PUF.

5.3 PUF building blocks

A PUF consists of a main block and an optional pre-processing block and post-processing block. The structure of a PUF is illustrated in Figure 2.

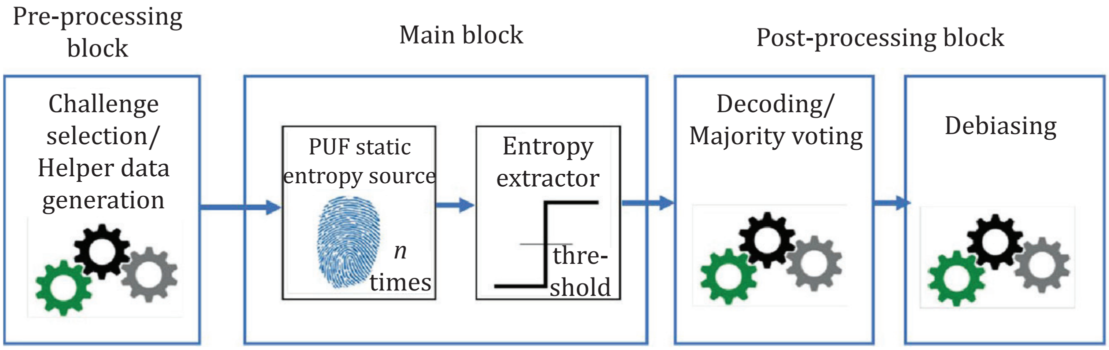

Figure 2 — Functional model of a PUF with analogue output measure from the static entropy source

The main block extracts the device variation and generates a unique output to the PUF. A main block consists of an entropy source and an entropy extractor. Some examples are given in Annex B.

The entropy source of a PUF is the device variation arisen during manufacturing process and so forth. The device variation is stationary when in use, and therefore the entropy source of a PUF is called a *static entropy source*.

The entropy extractor is a mechanism that digitize the analogue quantity generated from the entropy source.

Since the PUF responses are generated from the subtle device variation, the responses are sensitive to noise and often include erroneous bits. The optional pre-processing block is used to improve the steadiness of the responses.

One solution to improve the steadiness is to discard unstable CRPs. In Figure 2, to increase the steadiness, some function can be applied to the challenges, e.g. to mask unreliable elements amongst the "*n*" individual entropic elements making up the PUF. Another solution is to apply ECC to the responses. The pre-processing block generates helper data that is subsequently used to correct the error bits.

The optional post-processing block also improves the steadiness of the PUF responses. The decoding block applies majority voting or ECC decoding to correct the error bits in the responses.

Another purpose of the post-processing is to eliminate bias in the responses. The output distribution of a PUF can be equalized by removing bias by implementing a Boolean function with good diffusion properties[1].

Note that in Figure 2, the two optional blocks "decoding block" and "debias block" may, under certain circumstances, be merged or swapped (as in Von Neumann debiasing or Index Based Syndrome[2] debiasing). Because the post-processing can leak entropy, the post-processing should be carefully analysed.

5.4 Use cases of PUF

5.4.1 Security parameter generation

One usage consists in using PUF outputs as security parameters such as cryptographic keys, nonces, seeds for random number generators, etc. Figure 3 shows the example block diagram of the cryptographic key generation with a confined PUF.

While a non-volatile memory virtually permanently stores a secret, a confined PUF generates a volatile secret. To use a PUF for security parameter generation, the PUF is required to output an identical response to the given challenge with a small bit error.

An extensive PUF, which indeed has a challenge input, can be used equivalently as a confined PUF if it uses a constant challenge set.

Generally, a response of a PUF is sensitive to noise. Therefore, the error correction scheme called the fuzzy extractor is usually necessary to use a PUF for security parameter generation. The fuzzy extractor usually requires the redundant information called the helper data.

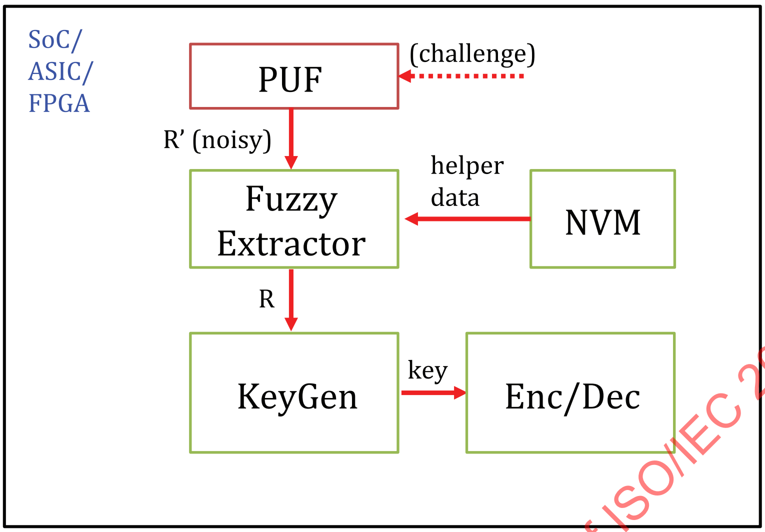

Figure 3 — Example of the security parameter generation with a confined PUF

5.4.2 Device identification

The second usage consists in device unfalsifiable identification. In this case, a PUF is used as a public identifier, which, therefore, shall not be cloned, nor falsified. An example of this use case is prevention of the intentional over-production of silicon chips. Without an unfalsifiable and unclonable identifier, a dishonest manufacturer can possibly overproduce chips and sell them as genuine. Manufacturing silicon chips with a PUF, the over-produced chips may not be recognized as genuine due to the unique and unfalsifiable identifier.

5.4.3 Device authentication

The third usage consists in authentication of a device. This application is demanding more security than the plain identification, as the protocol shall be protected against replay attacks. Therefore, in this use case, a subset of PUF's challenges and responses is saved and this whitelist enables future interactive attestation that the device is genuine. The scenario is referred to as challenge-response protocol.

Figure 4 shows the example of the PUF-based challenge-response authentication (hereinafter, simply a "challenge-response authentication") using an extensive PUF. The challenge-response authentication has two phases called the enrolment and verification.

In the enrolment phase, the verifier collects the CRPs of the PUFs to be verified and record them to a database. The CRPs are associated with the ID of the PUF. The enrolment is performed in a secure environment by a trusted player, e.g. a manufacturer, vendor and service provider.

After the enrolment, products equipped with a PUF are shipped and delivered to a user. Then the verification is performed in the field by request from a verifier. In this example, the communication channel is not encrypted, and hence the challenge, C, is supposed to be used only once to avoid a replay attack.

In the verification phase:

a) a user of the PUF sends the PUF ID to the verifier;

b) the verifier takes challenge, C, from the database and sends it to the user;

c) the user inputs the challenge, C, to the PUF and obtains the response, R';

d) the user sends the response, R', to the verifier;

e) the verifier collates the returned response, R', and recorded response, R, in the database;

f) if R = R', the PUF is authenticated.

Because responses of a PUF usually include erroneous bits, authentication is performed based on the fuzzy identification scheme with a threshold. In the fuzzy identification, a PUF is authenticated when the Hamming distance of R and R' is less than the set threshold. If the intra-HD and inter-HD of a PUF are separately distributed, the PUF is perfectly identifiable, and therefore authenticated. If the distribution of the intra-HD and inter-HD are overlapped, FAR and/or FRR is not zero, and therefore, identification error occurs (see Figure 5).

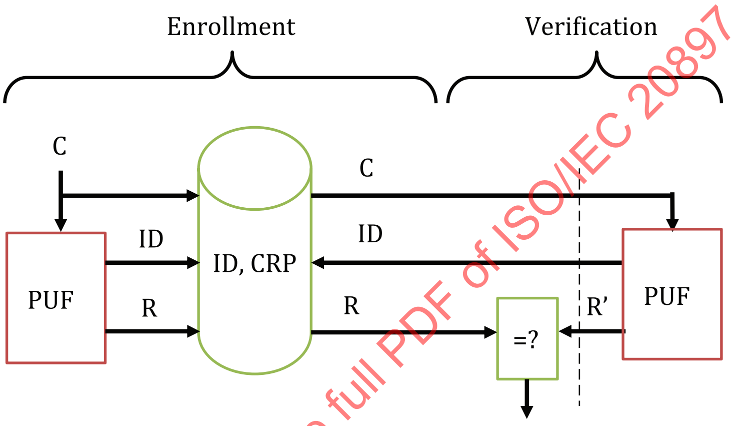

Figure 4 — Example of the challenge-response authentication with an extensive PUF

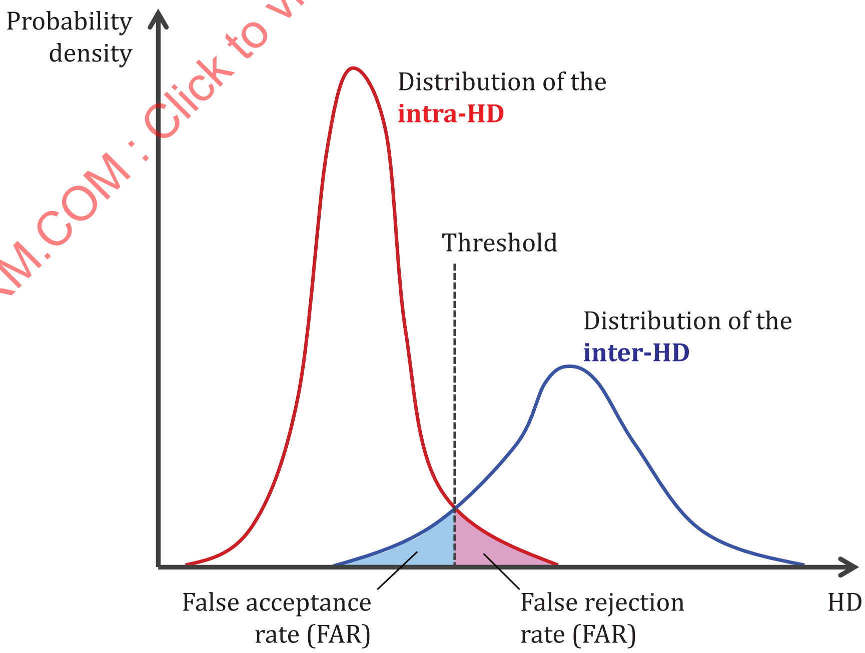

Figure 5 — Distributions of the intra-HD and inter-HD

5.5 Security requirements

5.5.1 General

This subclause defines the security requirements which a PUF shall satisfy for the security purposes: steadiness (5.5.2), randomness (5.5.3), uniqueness (5.5.4), tamper-resistance (5.5.5), mathematical unclonability (5.5.6) and physical unclonability (5.5.7).

For the purpose of this document, the responses from multiple PUFs are arranged into a cube as shown in Figure 6. The repetitive calls to a PUF are illustrated in Figure 7. The single small cube describes a 1-bit response from a PUF. The three axes of the cube and the time are described as directions:

- direction B: "#bits" shows the bit length of the response obtained from a single challenge. In a 1-bit response PUF, e.g. arbiter PUF, the dimension B collapses.
- direction C: "#challenges" shows the number of different challenges given to a PUF. In a no-challenge PUF (or, more rigorously, a one-challenge PUF), e.g. SRAM PUF, the dimension C collapses.
- direction D: "#PUF" shows the number of different PUF devices under test.
- direction T: "#query" shows the number of query iterations under the fixed PUF device and challenge.

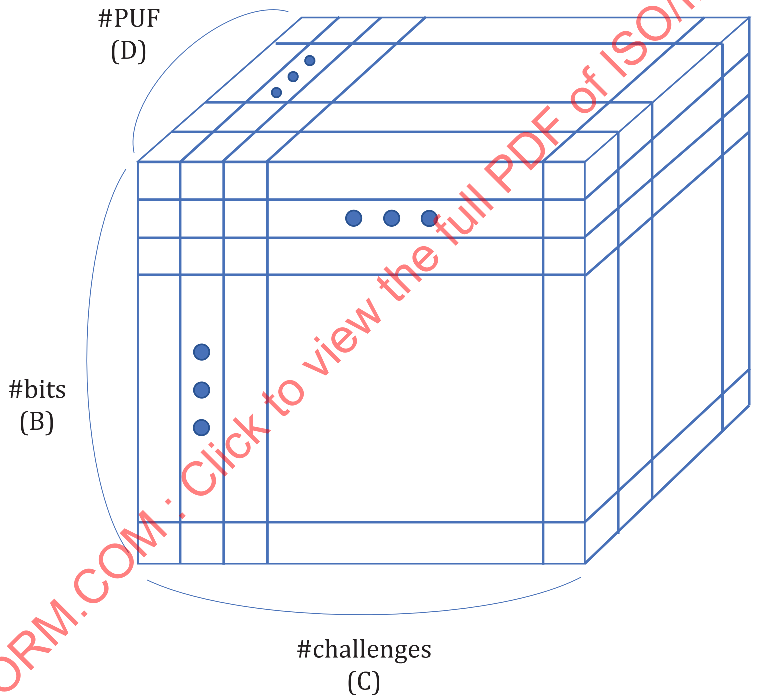

Figure 6 — Cube representation of the response sequences from multiple PUFs

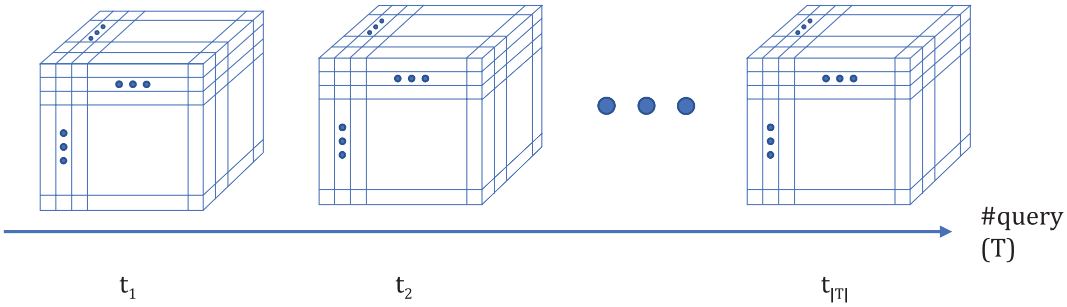

Figure 7 — Responses obtained by repetitive calls to the PUFs.

5.5.2 Steadiness

If a PUF is used for security parameter generation, device identification and device authentication, the PUF shall satisfy *steadiness*. Steadiness is the property that the response of the PUF is (almost) always the same if the same challenge is repeatedly given, irrespective of external manipulation (within the allowed environmental conditions specified by the PUF). External manipulations are:

- repetition of queries; either due to a method to increase the steadiness, or due to the reboot of the system, which requests the same response at each boot;
- variation of the environment within the range of allowed values.

5.5.3 Randomness

If a PUF is used for security parameter generation and device authentication, the PUF shall satisfy *randomness*. Randomness is interpreted as the entropy of the responses in the B-C plain in Figure 6.

5.5.4 Uniqueness

If a PUF is used for security parameter generation, device identification and device authentication, the PUF shall satisfy the *uniqueness*. Uniqueness is the inter-PUF property that for any two instances of the same PUF design, their responses are different. Uniqueness is interpreted as the entropy of the responses in direction D in Figure 6. This property arose from the intent to have the distribution of responses over the PUF instances to be uniform.

5.5.5 Tamper-resistance

The PUF responses shall not be read out or manipulated in unintended ways by invasive or non-invasive attacks. Such attacks include side channel, reverse-engineering, LVP, FIB and many other techniques borrowed from the failure analysis toolboxes. Tamper-resistance is the property that it is impractical for attackers to read out and/or manipulate the responses.

5.5.6 Mathematical unclonability

A PUF shall not be mathematically clonable. That is, the PUF's input-output behaviour shall not be simulated or emulated by other devices. Mathematical unclonability is the property that it is difficult to relate the challenge, design and architecture to the responses it generates. Mathematical unclonability related to the fact that a PUF shall not be easily copied to be implemented, e.g. in a simulated device with the same inputs/outputs functional behaviour.

For example, if all CPRs of a PUF are collected by an attacker, the attacker can build a complete mapping table of the CPRs and therefore the PUF is considered to be mathematically cloned.

In addition, if a mathematical model of the input-output behaviour a PUF is built by analysing some CPRs (also known as machine learning attack), the PUF is also considered to be mathematically cloned. Mathematical unclonability ensures that even if (for some reason) some pairs of challenge and corresponding responses are leaked, then an attacker cannot predict further responses to the challenges not seen before.

5.5.7 Physical unclonability

A PUF is considered to be physically cloned if, without building a mathematical model, another PUF is manufactured so that it behaves in the same way as the original one. A PUF shall not be physically cloned, even if the design of the PUF is stolen or restored by reverse engineered by an attacker. Physical unclonability is the property that it is difficult to fabricate two (or more) PUFs that behave the same.

5.6 Mapping between security requirements and use cases

It depends on the usage (see 5.4) and the life-cycle of the PUF (see Annex C) which security requirements the PUF shall meet. Table 1 underlines the mapping between the security requirements of a PUF and its applications.

Security parameter generation should be:

- steady, otherwise the security parameters cannot be reproduced;
- random, otherwise "key brute forcing attacks" become possible;
- unique, otherwise the keys of different devices become identical with high probability;
- tamper-resistant, otherwise keys will either be read-out or fixed to a chosen value;
- mathematically unclonable, otherwise "future keys will be predictable";
- physically unclonable, otherwise "pretending to be a genuine device (even without knowing the key)" becomes possible.

Device identification should be:

- steady, otherwise "unauthorized access" can become possible;
- unique, otherwise the IDs of different devices become identical with high probability;
- tamper-resistant, otherwise "IDs can be copied or forged";
- mathematically unclonable, otherwise "ID is no longer unique";
- physically unclonable, otherwise "device impersonation attacks" become possible.

Device identification may or may not satisfy randomness.

Device authentication should be:

- steady, otherwise genuine devices cannot be correctly authenticated (FRR increases);
- random, otherwise genuine devices can be impersonated;
- unique, otherwise other different devices can be wrongly authenticated (FAR increases);
- tamper-resistant, otherwise "authentication can be copied or forged";
- mathematically unclonable, otherwise the impersonation attack succeeds with high probability;
- physically unclonable, otherwise "device impersonation attacks" become possible.

It should be emphasized that the generation of deterministic keys (such as serial number, identification number, etc.) is more security demanding than ephemeral keys.

Threats depend on the operational environment, hence security requirements shall also be considered in a hostile manner. An example of PUF selection (including its security requirements) process is as follows.

- Analyze all security design with a set of tenants (e.g. the design is known, thus, it can be safely considered that all other information other than local variation is known).
- Then consider the application in the selection of the solution. Typically, the security of a system is highly dependent on the adversary’s access to the system, the value of what is being protected and the capabilities of the adversary.

Security requirements shall also be evaluated relative to other cryptanalysis methods aiming at recovering CSP, such as exhaustive search or differential fault analysis (DFA) when the CSP is subsequently used as a secret key in a block cipher.

Table 1 — Mapping between threats and security requirements

Security requirement/Application	Key generation	Identification	Authentication
Steadiness	√	√	√
Randomness	√		√
Uniqueness	√	√	√
Mathematical unclonability	√	√	√
Tamper-resistance	√	√	√
Physical unclonability	√	√	√

Annex A
(informative)

Classification of PUF

A.1 General

PUFs technologies can be diverse. It is important to interpret security requirements and tests within the context of each particular PUF technology. The purpose of this annex is to explicit some classification features.

A.2 Confined vs extensive

Refering to 3.2 and 3.5, a confined PUF can accept or not accept inputs, whereas an extensive PUF usually receives inputs called challenges. Therefore, extensive PUFs are amenable to producing larger entropies for a given implementation than confined PUFs.

A.3 Silicon vs non-silicon PUFs

A silicon PUF can be manufactured using the same technology as the rest of the system it is embedded in. A non-silicon PUF exploits the variation of an external non-silicon object, e.g. an externally placed optical media or the coating material over the chip surface.

A.4 Delay vs non-delay

The physical origin of randomness of a delay PUF is the race between electrical signals. Bits are derived from the comparison between the relative speed of signals. Non-delay PUFs exploit other hardware-specific physical phenomena as a source of physical entropy.

Annex B
(informative)

Some PUF implementations

B.1 General

Silicon PUFs may be integrated with other analogue and digital functions into an electronic circuit, thereby procuring an enhanced security level: the secrecy of the challenges and the responses is easier to guarantee since the ports of the PUF are not exposed.

Non-silicon PUF are not implemented as an electronic circuit. As most applications of PUFs require some data processing and transmission, the PUF is likely to be coupled with an electronic circuit, thereby making up a heterogeneous system.

B.2 Silicon PUF implementations

B.2.1 Arbiter PUF

Let one signal be split and run through two paths made up of several identical elements; the principle of the arbiter PUF is to accurately determine which signal is the fastest. Such information allows to build a security parameter on a bit-by-bit basis. For more detailed information on arbiter PUF, see Reference [3].

B.2.2 SRAM PUF

SRAM cells consist in metastable memory points. Metastability arises from the cells being bi-stable. Some cells are well balanced, thereby featuring a random behaviour. At the opposite, some cells are biased, and therefore tend to start always at the same state. Those cells constitute the support of the PUF unpredictable albeit deterministic security parameter. For more detailed information on SRAM PUF, see Reference [4].

B.2.3 Loop PUF

The loop PUF consists in one self-timed oscillating loop, which runs at a frequency which is challenge-dependent. The loop-PUF features a counter which is incremented upon each round in the oscillating loop. Relative comparison between frequencies allow to deduce the bits of the PUF. For more detailed information on loop PUF, see Reference [5].

B.2.4 Glitch PUF

Combinational logic consists of gates which can anticipate their computation: as soon as any input toggles, the output is updated. There is no synchronization of the inputs, therefore any combinational gate may evaluate several times. Such phenomenon is referred to as a glitch. Glitches occur owning to races in combinational gates. The more gates, the more glitches. Since glitches result from races, they are extremely prone to process variability. The sampling of glitching logic therefore make up interesting device signatures, which can be turned into a unique identifier. For more detailed information on glitch PUF, see Reference [6].

B.2.5 Clock PUF

Clock PUFs are arbiter PUFs where the two competing signals are not running along dedicated gates, but are signal propagations on clock trees, designed to provide (ideally) balanced. In practice, the trees