Job Nestor Bahner

Military Autonomous 5G-Based Systems: Using PUF for Hardware Authentication to IoT SAFE SIM

Masteroppgave i Master Information Security
Veileder: Stephen Dirk Bjørn Wolthusen
Juni 2023

Masteroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for informasjonssikkerhet og kommunikasjonsteknologi

Job Nestor Bahner

Military Autonomous 5G-Based Systems: Using PUF for Hardware Authentication to IoT SAFE SIM

Masteroppgave i Master Information Security
Veileder: Stephen Dirk Bjørn Wolthusen
Juni 2023

Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for informasjonssikkerhet og kommunikasjonsteknologi

Job N. Bahner

Abstract

The increasing use of autonomous systems has interested the Norwegian armed forces. In order to facilitate this, as well as their own internal communication in the case of an internet blackout they have looked to 5G, as a supporting technology. A possible problem that arises when connecting many devices, to an internet, as well as the internet is how to make sure that all devices are authenticated properly. In a normal setting, this is mostly solved, but the military context requires additional measures to ensure the safety of their devices, as many small autonomous systems cannot be physically guarded, as is standard in the military. You cannot send military guards to protect a sensor in the forest, for example. For now, the underlying authentication infrastructure is in place, but they still have a problem if someone were to steal the SIM cards that they use for authentication. These SIM cards are based on the IoT SAFE (SIM Applet for Secure End-to- End Communication) standard, and are in and of themselves safe, but is agnostic to the device it is in. In order to make sure that only certain devices can be authenticated with certain SIM-cards, there is speculation that this can be achieved by using Physically Uncloneable Functions (PUFs) from for example the SRAM. Research into this was done, and the requirements for secure, reliable, and robust authentication with PUFs are defined. This is done through a series of measurements of commercially available SRAM chips, and defining an authentication system based on the opportunities and limitations provided by the technology at hand.

Sammendrag

Økende bruk av autonome systemer har vakt interesse hos de norske væpnede styrkene. For å lette dette, samt deres egen interne kommunikasjon i tilfelle av en internett-nedetid, har de sett på 5G som en støtteteknologi. Et mulig problem som oppstår når man kobler mange enheter til internett, er hvordan man sikrer at alle enheter blir riktig autentisert. I en normal setting er dette stort sett løst, men i militær sammenheng kreves det ekstra tiltak for å sikre enhetenes sikkerhet, da mange små autonome systemer ikke kan være fysisk bevoktet, slik det er vanlig i militæret. Du kan for eksempel ikke sende militære vakter for å beskytte en sensor i skogen. For øyeblikket er den underliggende autentiseringsinfrastrukturen på plass, men de har fortsatt et problem hvis noen skulle stjele SIM-kortene de bruker for autentisering. Disse SIM-kortene er basert på IoT SAFE (SIM Applet for Secure End-to-End Communication) standarden, og er i seg selv sikre, men er agnostiske i forhold til enheten de er i. For å sikre at bare visse enheter kan autentiseres med visse SIM-kort, spekuleres det i om dette kan oppnås ved å bruke Fysisk Ukopierbare Funksjoner (PUF) fra for eksempel SRAM. Forskning på dette er gjort, og kravene for sikker, pålitelig og robust autentisering med PUF-er er definert. Dette gjøres gjennom en serie målinger av kommersielt tilgjengelige SRAM-brikker, og ved å definere et autentiseringssystem basert på mulighetsrommet og begrensningene til teknologien som er tilgjengelig.

Contents

Figures

Code Listings

Chapter 1

Introduction

As the Internet of Things (IoT) is being innovated at a high rate, so is its adoption in many areas of industry, and in private applications. Security - which has a comparably slow rate of innovation, is lagging behind. This causes an increase in the *attack surface* of which the internet has not seen since its very early stages. This concerns professionals in the field, as well as businesses, and national security interests - which are more exposed; this is naturally due to critical infrastructures being a concern of national security. Another domain in which IoT technologies are being employed, is in the defence sector. The defence sector wishes to utilise new 5G technologies to improve their standalone communications capabilities in the case of an attack on national communications infrastructure. The defence sector also wishes to see the future of defence becoming more autonomous, thus risking fewer lives. 5G technologies enable this.

The modern military is changing. There is a move towards a kinds of warfare and defence that relies to a greater extent on technology, and less on people. The defence forces have always been early adopters of high-technological approaches, and 5G is, as of this time, a high-technological approach to the threat of warfare on human lives, and the citizens of which the military seek to defend. In order to ensure a robust defence, and secure the state it is of paramount importance to make sure that the defence forces keep functioning even during a siege. This is also an application of 5G edge computing, and using dedicated parts of public networks to ensure resilience in such a situation. The problem with increasing automation is the lack of direct contact with the nodes that are performing their functions. It is also important to point out that with an increasing reliance on technology, there must also be an in crease in trust and resilience in these products. An approach to secure these devices is to make the devices prove to the other devices on the network that the device is "who" it is to the other nodes on the network. This is to mitigate the impersonation of devices to perform attacks, and interception of sensitive communications. To address this problem a standard by GSMA, called "IoT SAFE (SIM Applet for Secure End-to-End Communication)" has been created to authenticate devices to mobile (GSM) networks, which is cryptographically secure and tamper proof. This is all fine and well, but it does not address what happens if such a SIM card were to be stolen. This is where pairing the SIM card with a specific device can be of good use. If that is done, then the SIM card cannot be used with any other devices than the one it is intended for. This does not mitigate theft of the device, which is out of the scope of this thesis. To solve the usage of such device+SIM authentication, the use of a Physically Uncloneable Function PUF is an option. A PUF is a way of using unique manufacturing differences, in for example the RAM to uniquely identify a device. In order to see if this is a feasible approach, the author will seek to understand how IoT SAFE, and PUFs work. Subsequently there will be a look into how to actually implement this, and how to address the various challenges posed by the project commissioner, Thales Group Norway Inc. To actually implement this the author will use his background in information security technology, and his knowledge in hardware security to study the theoretical and practical facets of using PUFs create a composite key with IoT SAFE.

1.1 Motivation

The motivation for this thesis is to enable secure and effective communication of IoT devices in the defence sector; in order to prevent breaches of integrity and confidentiality. The issue at hand comes to light when looking at the current methods available for IoT authentication, which were not designed to be used in such environments, much less in the defence sector. These methods include protocols such as TLS, and different PKI scehmes, which rely on an assumption of secure identity provisioning, and secure data management. Since many IoT devices are constrained in computing power, and have strong requirements for both chip-space, and power consumption the prospect of limiting the resource usage of certain cryptographic elements is highly interesting. Especially since security in many IoT application is a mere sideline to their actual purpose an can be seen as an impedance, and thus loose standing in the prioritisation queue. For this reason, non-compromising alternatives to strong cryptography is appealing.

In the military context, in which Thales Norway operates, they recognise that other issues also arise [1]. Specifically related to the upcoming shift to an autonomous military 5G infrastructure - SIM cards. SIM cards are used to authenticate to the telecommunications infrastructure which provides high-speed, low-latency networking for field-appliances. These appliances may need real-time support, and often have very high security requirements in naturally adversarial environments. The question "What happens if someone were to obtain physical access to field devices, and spoofs their data-stream?" arises as a caveat to this physical setup. Along with "How can a read-off-chip in an off powered state be prevented?". The answer to both of these questions may be PUFs. As PUFs cannot be read off-chip before after the device has booted (side-channel attacks aside), and a weak PUF may be used as a secure input to a composite key-scheme where a PUF is used to intrinsically link a device to a SIM-card, and a SIM-card to a device. In this instance, the SIM-card can be remotely revoked, rendering the entire combination useless. Thus, the motivation is clear. PUFs may solve the next-generation challenges posed by the military IoT domain. Another engaging factor is that this research that will be used to determine the feasibility of such application in a real product being developed by Thales Norway, which ends up aiding the Norwegian defence forces in the case of an attack or breach. This is of high societal value.

1.2 Goals

The goals for this thesis is to assess the state-of-the-art in PUF-technologies, assess the state of IoT SAFE, and to see how the inter operation of these can be used to authenticate end-nodes in networks in general, and military field networks in particular. In addition to assessing how well the proposed scheme works, in comparison to the power consumption of non-PUF methods, as well as measuring the amount of requests that are falsely denied (false rejection rate FRR).

1.3 Research Questions

Based on the literature, discussions with the project supervisor, and Thales Norge, the following research questions are posed:

- What are the theoretical applications and limitations of using PUFs for autonomous systems?
- Will the use of a PUF significantly improve the ease an security of autonomous military systems, from the theft of SIM-Cards with IoT SAFE?
- How reliable are the methods presented for use in authentication, in terms of statistical repeatability, and changing environmental conditions? Including power consumption.

1.4 Problem Description

Certain use cases for the defence sector, requires devices to be deployed in a field environment, in which devices are physically exposed. In order to provide authentication to these devices, and to ensure the confidentiality and integrity of their communication, the defence researchers are looking into the use of IoT and 5G enabling security technologies, such as IoT *SIM Applet for Secure End-to-End Communication (SAFE)* to protect their field-deployed devices. A potential *attack scenario* to these devices, is theft of the device, or its SIM-card.

In this case, the field device could easily be spoofed, and thusly loss of life could occur as a result of faulty decision making. To protect against this, it has been theorised that a composite key could be created from a unique property of each and every device, akin to a fingerprint. This would enable the device to be invariably linked to the SIM card, so that a separation of the two is not possible. This technology is called a *Physically Uncloneable Function (PUF)* and has been broadly researched over the course of the last 11 years. The aim of this thesis is to understand how these technologies can be used in a practical setting, and to assess which meaningful advantages PUFs possess in relation to traditional methods, using non-PUF based authentication in terms of power-consumption.

Whilst the issues of information-spoofing and unnecessary power-draw may be critical in the systems which fall under the umbrella of tactical networks, the case, may very well be that the implementation of PUFs in such environments may not provide any significant benefits, and instead create more esoteric systems that do not scale well, and, of course not achieve the desired effects of improved physical security and lessened resource consumption.

1.5 Planned Contribution

This thesis aims to create a functional understanding of the practical uses of PUFs in advancing IoT security for defence use cases, in a highly dynamic 5G environment. Thus, the thesis seeks to understand the current state of the interactions between PUFs and IoT SAFE in dynamic networks (such as private-5G). The aim is to secure IoT devices in these environments by means of a practical understanding of component interactions, so that Thales Norge AS can deliver a product which will advance the "5G-VINNI" project, and consequentially, the state of the Norwegian Defence 5G infrastructure and capabilities.

The planned contribution is therefore to assess the usefulness of PUF technologies in authentication, in comparison to traditional methods where the keys are stored in NVM (Non Volatile Memory), and can be extracted by an attacker with physical access. One may think that this would simply move the problem to another area of the chip, and not fundamentally presenting a solution, but it does, in theory create an opportunity to improve on the current issues which are being faced in this domain.

1.6 Thesis Structure

This thesis is structured into three main parts. The introductory chapters, 1 Introduction, 2 Background, and 3 Methodology. The main part consisting of the chapters, **??** Analysis, 4 Design, and 6 Discussion. The last part consists of the 7 Conclusion, and the chapter **??** bibliography and appendices.

The introductory chapters seek to inform the reader about the purpose, scope, goals, motivation, outline and shape of this thesis, as well as setting the scene for the rest of the thesis using relevant background materials needed to understand the subsequent reading. The methodology is also encompassed within the scope of the introductory chapters, and seeks to outline how exactly the thesis is created, upon which ideals and bases its founded upon, as well as establishing some key assertions regarding ethics, timeline, feasibility, and risk.

Following these introductory chapters, follow the main chapters which aim to understand and apply the background material in relation to the goals set in the introductory chapters, using the background, relevant literature, and the methods described in chapter 3. These chapters seek to apply the source material in such a way as to create a strong foundation for the discussion chapter, where the data and background are used to answer the research questions as well as possible, in order to create the basis for the design of the system.

Following this the concluding chapters seek to wrap up the findings from earlier in the paper, as well as establish what exactly is now known, and what is still left unknown, in the form of future research endeavours. Here the relevant background literature is also listed in the form of a bibliography, and appendices.

Chapter 2

Related Work

2.1 Background

This chapter aims to introduce the concepts that are defining for this thesis. It will cover the areas of PUFs, IoT SAFE, Military Networks (as opposed to conventional networks), 5G, End-node Authentication in Military Networks, Taxonomies and PUF Definitions, Statistical Modelling (as used in this thesis), and a reasonably deep dive into probability density functions.

2.1.1 PUF

According to Maes [2] a PUF is a *Physically Unclonable (Probabilistic) Function* as its primary function is that is is a physical property of an object, that is often intrinsic to the manufacturing process, from inevitable production variations, that cannot be cloned or repeated. When discussing these physically unclonable functions, it is of a large interest to be able to describe them accurately so that one can discern what is and is not a PUF, as to not get them confused with similar concepts, such as POK (Physically Obfuscated Keys)s, or hardware keys in a separate memory location. As these are non-intrinsic, i.e added purposefully during manufacturing, much like a key card is given to an employee, whilst a PUF is more akin to a fingerprint. Regarding the different variants of PUFs, there have been proposed a large variety of models to perform the intentions expected by a PUF.

As a PUF is intrinsic to the device, it has been described in literature as an identifying feature of an entity. And authors, thus argue, such as [3] which uses identification as a synonym to authentication, whilst more stringent definitions would argue that this definition is indeed problematic, as an identity is a proposal of a purported identity, requiring no formal verification of the purported identity. Authentication on the other hand, does require a formal verification of the credentials of the purported identity in order to ascertain the real identity of the authenticating entity.

There are also other reasons for identification and authentication being inherently different. For one, most if not all authentication schemes rely on an identity as a precondition for authentication, secondly, for some systems identification alone is enough. Lastly, identification can be enough to authenticate an entity, as the preconditions for authentication can be implicitly met.

There are various kinds of PUFs that have been studied, and that have been measured towards each other. Maes [2] identified seven main PUF constructions, all with unique properties, and subsequently evaluated these to find which ones had the better inter and intra distances. As well as their uniqueness and reproducibility in a variety of temperatures. After this seminal work, few new approaches have been proposed, but the old ones have been expanded upon. These are: Arbiter, ring oscillator, glitch, SRAM, Latch flip-flop buskeeper, and bistable ring PUFs. More recent and novel PUFs include the quantum-dot cellular automata (QCA), ReRAM, Mirror-Circuit, IOE, Soft-BD, Quantum Tunneling, CRC, Mc, DRAM, and MUX PUFs.

From the experimental results derived from the work of [2], the author derives that Maes asserts that PUFs based on SRAM constructs seem to be the most reliable in terms of intra and inter distances, as well as using very little space on a chip, as well as using practically no more power than absolutely necessary, as the SRAM PUF is used during the start up routine of the device. If this PUF is stored in memory, after initialisation then the PUF can be used "on-the-go", and the device would not have to be restarted to re-present the key. There are also measures that have been taken to assure the stability and reproducibility of PUFs, such as using hashing algorithms, error-correcting algorithms, and prevent model building attacks. Another feature of using hashing algorithms is that they can be combined with secondary input to create unique keys for various accounts on the system. For these reasons, it could be feasible to use a PUF for more than a single root of trust, but also as a seed for a random number generator, or as a similar source of entropy.

PUFs are measured using a variety of statistical metrics, which determine how random and unpredictable they are, between each other, and in relation to itself. These metrics are known as *intra* and *inter*-distances for the PUF in question. Whereas an optimal intra distance of a PUF is exactly 50%. as measured by the fractional hemming distance as is elaborated on in **??**. The intra distance is best seen at 0, for 0 variation between each measurement, meaning it stays consistent, and that the PUF is inherent. Thus, an optimal PUF result, should be both random, but also reproducible for each separate entity.

Formally speaking, PUFs can be divided into certain groups based on their class, and their instance. a *PUF class*

$$\rho$$

, is a "blueprint" of a PUF which contains the features of the exact PUF, which then can be instantiated as a *PUF instance*, which is the actual implementation. The class can be characterised as a set of all its created instances, according to [2] and is characterised thusly,

$$\rho\{puf \cong \rho.Create((r^C)_i) \implies puf_i : \forall, \{0.1\} * \$ \implies (r^C)_i)\} \tag{2.1}$$

with

$$r^C \tag{2.2}$$

representing a finite amount of random outcomes. A PUF instance puf, can also be evaluated by means of a physical experiment which yields the set

$$\gamma_\rho \tag{2.3}$$

of all physical instances of the puf class. These instances can be evaluated, which results in a measurement of the physical state of

$$puf_I \in \rho$$

. As PUFs reflect a set of physical properties of an object, they are inherently affected by physical parameters *conditions*, such as temperatures, voltages, magnetic interference etc... Such conditions, other than nominal are denoted as

$$Eval^\alpha$$

Taxonomy

PUFs have gone through multiple iterations in research, and has had waves of interests and applications. For this reason different requirements have emerged throughout the years. Starting from certain use cases that were thought to be the most relevant, to different approaches that set different standards for the designs and criteria relating to PUFs. The main divide between PUFs, are first and foremost their inter and intra-distance metrics, which determine the overall viability of the specific PUF technology. The second contention is the area of application and use for the selected PUF technologies. This is based both on the proposed applications of the PUFs, and the more realistic applications which the PUFs can *actually* fulfil.

PUFs can be divided into two primary categories: weak and strong PUFs, despite their seemingly contradictory names in terms of actual strength. In fact, weak PUFs, paradoxically, display greater strength than strong PUFs, as mentioned in the citation [insert citation here]. Essentially, weak PUFs serve as an innovative means of securely storing secret keys in vulnerable hardware, providing an alternative to ROM, Flash, or other non-volatile memories (NVMs). Like all PUFs, weak PUFs possess an inherent and unclonable physical disorder within them, leveraging a challenge-response mechanism to exploit this disorder. Furthermore, the following are the characteristic features of weak PUFs:

1. Few challenges: A Weak PUF has got very few, fixed challenges, commonly only one challenge per PUF instance.
2. Access-restricted responses: In all but very few applications, the challenge-response interface (or the challenge-response mechanism, respectively) of a Weak PUF needs to be access-restricted. It is assumed that adversaries cannot access the Weak PUF's responses, even if they hold physical possession of the PUF-carrying hardware.

So-called "Strong PUFs" are the second major PUF type besides Weak PUFs. In opposition to the latter, they derive a more complex challenge-response behaviour from the physical disorder present in the PUF. Typically, many physical components are involved in the generation of a response, and there is a very large number of possible challenges that can be applied to the PUF. In a nutshell, they can be subsumed as follows:

1. Many challenges: Strong PUFs have a very large number of possible challenges, ideally (but not necessarily) exponentially many challenges in some system parameter. This prevents a full read-out of all CRPs, even if an adversary holds physical possession of the PUF for considerable time.
2. Unpredictability: Even if an adversary knows a large subset of CRPs, he cannot extrapolate or predict the other, yet unknown CRPs.
3. Unprotected challenge-response interface: In all but very few applications of Strong PUFs, it is assumed that they have a freely, publicly accessible challenge-response interface (or a freely accessible challenge-response mechanism, respectively). Anyone holding physical possession of the PUF or the PUF-carrying hardware can apply arbitrary challenges to the Strong PUF and read out the corresponding responses.

2.1.2 Weak and strong PUFs

PUFs possess a distinguishing characteristic known as implementation strength, which is categorised into two levels: weak and strong. The strength of a PUF is determined by the quantity of challenge-response pairs (CRPs) that can be generated from a single device. Typically, this corresponds to how the number of CRPs increases with device size, serving as a metric for assessing PUF strength. However, it should be noted that there are exceptions to this scaling rate, which will be discussed later in this chapter.
Weak PUFs support a relatively limited number of CRPs due to a low-order scaling rate. Consequently, if an attacker gains physical access to the PUF for any period of time, they can read the complete set of these pairs from the device. Although it is not possible to replicate the physical PUF itself, possessing knowledge of the PUF's CRPs allows an attacker to convincingly respond to queries as if they still had access to the device, even long after it has been removed from their possession. Weak PUFs can be utilised for secure key storage and entity authentication techniques, such as the protocol featured in [insert reference]. However, for authentication purposes, the PUF must be evaluated in an environment where an

authenticating party is present to ensure that the PUF itself is being accurately assessed.

A straightforward implementation of the weak PUF ensures that any attempted counterfeit response would noticeably differ from the recorded response of the genuine PUF. This detectable difference serves as a security measure against fraudulent replication.

In contrast, strong PUFs are designed to scale in a way that supports a significantly larger set of CRPs. The number of these pairs is so extensive that even if an attacker gains access to the PUF, it is practically impossible for them to record all the CRPs. During the manufacturing stage, if a random sample of these CRPs is selected, the chances of the attacker also capturing the response to the same challenge can be considered negligible. Consequently, the system ensures that only the user with physical access to the PUF at the time of the challenge can provide the correct response and be successfully authenticated. Additionally, the extensive repertoire of CRPs means that each challenge-response pair needs to be used only once, which prevents attackers from eavesdropping and enables the facilitation of secure communication protocols using the PUF.

In a simple implementation of the strong PUF, even if the PUF is compromised, an attacker would not possess knowledge of the relevant challenges to record. Furthermore, an eavesdropper would never be able to intercept a usable challenge-response pair. This robustness enhances the overall security of the system.

The strength of a Physical Unclonable Function (PUF) is typically assessed based on the scaling behaviour of the number of potential challenge-response pairs (CRPs) with the increasing size of the PUF. Generally, PUFs exhibiting exponential growth in the number of CRPs are considered strong, while those with linear or polynomial increases are typically categorised as weak PUFs. Strong PUFs yield significantly larger CRP sets as the PUF size increases.

Occasionally, PUFs with linear or high-order polynomial CRP growth are also referred to as strong PUFs. This classification may stem from factors such as limited read speed, high information density, or the ability to manufacture the PUF in parallel, resulting in a linear or polynomial increase in CRPs. For example, a PUF with exceptionally high information content may facilitate a redundantly large number of CRPs, akin to exponential scaling PUFs. It is important to note that constructing a true strong PUF, which is impervious to modelling attacks, is a highly challenging or even unattainable task. In such cases, techniques involving machine learning or physical intuition can be employed to extrapolate responses based on an attacker's limited set of obtainable challenge-response pairs. These techniques aim to derive computational rules for calculating responses to additional challenges.

Furthermore, the definition of a PUF itself remains subject to academic debate, as scholars hold differing views on the precise characteristics and criteria necessary for classifying a PUF. The determination of what constitutes a "weak" or "strong" PUF is also an area of disagreement within the academic community.

PUF architectures can be distinguished based on how the randomness, which is crucial for generating uniqueness, is incorporated. This distinction can be categorised as either explicit randomisation or implicit randomisation[4] [5].

Explicit randomisation involves the introduction of external steps to add randomness to the PUF architecture. For example, in CMOS electronic PUFs, additional CMOS components can be incorporated into the circuit deliberately, without the need for extra randomisation steps. The variation in these components arising from manufacturing processes contributes to the source of implicit randomness. On the other hand, if non-CMOS components are attached to a CMOS circuit, it would be considered explicit randomisation.

In general, implicit randomness, which naturally emerges from manufacturing variations, is preferred over explicit randomness in PUFs. Implicit randomness does not require additional processing steps, thereby avoiding additional costs. Moreover, the inherent process variations in typical manufacturing processes contribute to the implicit variation, which cannot be directly manipulated. Consequently, even the device fabricator cannot manipulate the manufacturing process in a way that removes or alters the random characteristics of the PUF. This ensures the integrity and reliability of the PUF's randomness.

In addition to the distinction between PUFs with implicit or explicit sources of randomness, PUF devices can be further classified into intrinsic and non-intrinsic evaluation varieties. An intrinsic PUF possesses randomness that arises in an implicit manner and also incorporates an internal evaluation mechanism. This means that the measurement or probing of the PUF is embedded within the device itself, making it an intrinsic property. Conversely, if a PUF does not meet these criteria, such as when it relies on a non-integrated implicit randomness source or an explicit randomness source, it is referred to as non-intrinsic or extrinsic.

Currently, the intrinsic property can only be exhibited by all-electronic PUFs since the only way to evaluate and obtain an electronic readout from a PUF is through an all-electronic evaluation mechanism. Internal evaluation mechanisms are more desirable than external evaluation methods as they allow for further processing, such as hashing, to occur without exposing the initial PUF response to the external environment of the PUF's internal circuitry. This integration of the randomness source and evaluation circuitry greatly enhances the resistance against man-in-the-middle and side-channel attacks between the two elements. Evaluation mechanisms internal to the PUF also tend to offer higher accuracy, ease of use, and reduced susceptibility to malicious interference.

PUFs have a wide range of applications beyond authentication. The weak PUF can be utilised as a means to generate and store a single or small number of cryptographic keys during manufacture. These keys can then be compared to an external database for identification or authentication purposes, or incorporated into other protocols such as secure communication or memory encryption. However, since the number of stored keys is limited, if an attacker gains access to the PUF and determines these keys, the system becomes compromised, similar to discovering a password or key in a conventional system.

The strong PUF can also serve the same applications and is capable of generating a large number of keys during manufacture for subsequent storage. Similar to the strong PUF authentication protocol, this allows for the redundant use of keys, enhancing security. This concept operates akin to a one-time pad in traditional cryptography, where each authentication exchange, secure communication message, or encrypted data bit can utilise a different key. Consequently, the compromise of a single key does not necessarily impact the entire system. Moreover, if the key is randomly chosen from the extensive set of possibilities, access to the PUF must occur simultaneously with the authentication, communication, or decryption process, as determining the specific key required for recording and replaying ahead of time would be infeasible.

In addition to the aforementioned applications, specific protocols have been developed to enable bit commitment, oblivious transfer, and secure key exchanges using PUFs. Certain PUF designs involve enclosing the PUF evaluation and/or other critical components within the source of entropy itself, forming an enclosure PUF system. These systems can be electronically evaluated, such as in the case of coating PUFs, or non-electronically evaluated, such as nanoparticle distribution PUFs with an optical evaluation method. The value of this approach lies in its tamper-evident nature, where any attempt to physically access or probe the PUF would disrupt the source of entropy and alter the PUF readout during subsequent evaluations. This feature can be instrumental in preventing side-channel attacks on the PUF's electronics or even rendering memory void or other circuits ineffective in the event of an enclosure breach.

There are additional extensions to the concept of PUFs that enhance their functionality and capabilities, namely the re-configurable PUF (rPUF) and the public PUF (PPUF).

The rPUF is a device capable of intentionally changing its response to the same input challenge. This enables the update of new challenge-response pairs and the revocation of previous CRPs, allowing the PUF to be reset for different purposes or users. Care must be taken to ensure that the newly generated response, reset in the field, remains unique and unpredictable, similar to the responses generated during the manufacturing phase. One example of an rPUF mechanism involves melting optical media in an optical PUF or controlled refreshing of a cell in non-volatile memory, such as PCM (Phase Change Memory) RAM or STT-MRAM (Spin-Transfer-Torque Magnetic Random Access Memory) PUFs.

The PPUF, also known as the SIMPL system, is more complex. In this case, the PUF can be modelled using parameters, although the process is time-consuming. This means that given sufficient time, someone with access to the parameters can derive one or more challenge-response pairs from the physical PUF. However, the time required for this process makes it infeasible to characterise a large number of CRPs, thereby rendering a full characterisation of the PUF impractical. On the other hand, the owner of the physical PPUF can easily obtain the corresponding response by challenging the device with little time burden.

2.1.3 IoT SAFE

IoT (Internet of Things) SAFE (Sim Applet for Secure End-2-End Communication) is designed to enable IoT devices to communicate securely, using a secure, physical root of trust [6]. Much work has been proposed for how to securely authenticate, authorise, and provide confidentiality for IoT devices, which according to Karie et.al [7] is the foremost challenge to IoT, currently. Studies from Business Insider asked executives what their greatest concerns regarding IoT were [8]. 39% of executives established security as the single greatest challenge in IoT. Still, safety in IoT is being widely studied, for all applications. This ranges from Cyber Physical Systems (CPS), Military systems, and networks, to home healthcare, and smart homes. There are many challenges that could be and, are being addressed. A key concern is, however, how to trust devices. Trust in IoT is difficult as the devices often are exposed physically, and are constrained both in terms of physical capacity, and network capacity for more remote slow-link devices. This is "forces" vendors to trade off security for their preferred level of functionality, often leaving users and end-devices, and ultimately networks at risk, as defences are only as strong as their weakest link.

In order to trust devices, which are bound to become more present in the era of 5G, GSMA has proposed a physical standard in the format of a SIM-card which would serve as a physical root of trust. This would explicitly establish trust in the end-devices, as they have a secure "hardware key" which can be used to establish a secure communications channel, providing both integrity and confidentiality, using a well-established and relatively light-weight protocol (D)TLS.

In military networks, however, there are different challenges to be addressed in order to create a secure 5G military backbone that would be used in the case of an attack on telecommunications providers, and the severing of national communications infrastructure. If the military sector is to transition to modern technology, and even futuristic technologies, such as autonomous vehicles etc. then they need to be completely sure that they can be trusted. IoT SAFE is a good option to perform this task, but does not fulfil the full range of threat scenarios faced by the armed forces. An example is an adversary stealing a SIM-card which is found in a sensor, and using this to send fake data to the owner of the sensor. In order to prevent this, a composite key which is intrinsically links the SIM-card and the device in which it resides is a way to mitigate this attack scenario.

From the implementation details, described in documents from GSMA, provide more insight into the thoughts and details behind IoT SAFE. They claim that there are challenges to current secure, scaleable, and manageable IoT secure authentication and authorisation [9], as well as enhancing the current security of IoT services [10]. Their whitepaper on IoT SAFE [9] describes how the protocol and the (e)UICC should function, and is described in the format of a *Request for Comment (RFC)*, whilst still being in its early stages. IoT SAFE is to be a SIM applet deployed as a *JavaCard* applet which provides (D)TLS functionality to an application wishing to establish such a session, by providing secure cryptography and the secure storage of either *Pre-Shared Keys (PSK)* or private keys.

2.1.4 Networks

A computer network is a way to create connections between devices, over digital media. In such communications, there are security challenges which could disrupt the integrity, confidentiality, availability, and non-repudiation of the contents of the network communication. The following section describes some key features of the networks relevant to this thesis.

Tactical Networks

Understandably, the armed forces are particular about their communications, and surly wish for their interactions, communications, and messages to stay private. The armed forces, are also prone to unique scenarios in terms of types of networks they use, but also their attack surface. With this it is meant that they may deploy highly mobile networks, such as field devices in ground-based warfare, boats, drones, sensors etc... Whilst at the same time having highly capable and persistent adversaries in opposition, i.e nation states, or *Advanced Persistent Threats (APT)s*. The armed forces are therefore in a particularly constrained environment with strict security requirements. Thus, they tend to develop much of their own technology for their very particular use case.

What makes these networks special, are

5G

5G, or 5th Generation, refers to the 5th generation of mobile networks, promising higher speeds, lower latencies, and large potentials in terms of automation. This technological leap, will enable many organisations to advance particularly quickly in terms of flexibility. The armed forces is one of these organisations, who has determined that 5G can be of great use for them. In this endeavour, the future use of autonomous vehicles, and speedy field networks is soon becoming a hard reality for the armed forces, advancing further into the next generation of warfare, in the electronic domain, rather than ballistic. Albeit, the use cases for ballistic operations is still present, but would largely be supported by autonomous entities, thus putting fewer humans at risk of injury or loss of life.

While 5G opens up a wide array of possibilities, the IoT domain still has to play catch up in order to reach utilise the potential of 5G. Whilst 5G promises higher bandwidths, more traditional internet protocols will most certainly be used, but the IoT constraints of battery life, and physical exposure are still present and need to be taken seriously if IoT is to be as widely adopted as technology-enthusiasts would have it.

Authentication of End-Nodes in Ad-hoc Networks

Whilst more traditional network authentication schemes may be used in 5G networks, field networks may very well have a long way to go when concerned with being decentralised, mobile, and operating on resource constrained devices. For this reason, there is an entire research field which is determined to find appropriate solutions to authenticate nodes in such isolated, and esoteric networks.

2.1.5 Statistical Modelling

When concerned with PUFs, there is a large focus on the statistical properties of the PUF. This is for the reason of gathering quantitative data on the properties of the PUF in terms of repeatability, reliability, randomness, and use in cryptographic operations in general. A subset of statistics will be used in this thesis to appropriately quantify and describe the properties which the PUF(s) exhibit. The most pressing statistical properties are described in 2.1.1 [11] such as *inter*,- and *intra* distances. There are other statistical properties which are also reasonable to define, as they will aide in describing PUFs accurately, so as to be used for the purpose of being intertwined with the cryptographic properties of the IoT SAFE (e, i)UICC. The statistical properties which will be used, and subsequently defined are seen in the following paragraphs:

Intra-Distance Definition

In regards to PUFs, the intra distance describing the distance between two responses from the same puf instance $puf_i \in \rho$ formally

$$\mathbf{ID}(x) \triangleq |\mathbf{HD}(Eval^{\alpha}(puf_i(x)), Eval^{\alpha}(puf_i'(x)))|$$

or simply

$$\mathbf{ID}(x) \triangleq \mathbf{dist}[Y_i(x), Y_i'(x)]$$

where *dist* is practically always *HD* or *FHD*, and x is a challenge.

Inter-Distance Definition

In regards to PUFs, the intra distance describing the distance between two responses from different puf instances $puf_{i,j} \in \rho$ formally

$$\mathbf{ID}(x) \triangleq |\mathbf{HD}(Eval^{\alpha}(puf_i(x)), Eval^{\alpha}(puf_j(x)))|$$

or simply

$$\mathbf{ID}(x) \triangleq \mathbf{dist}[Y_i(x), Y_i'(x)]$$

where *dist* is practically always *HD* or *FHD*, and x is a challenge.

Variables and Sets

A discrete random variable is defined as Y, a particular outcome of Y is denoted as its lowercase variant Y = y, the set of all outcomes of Y is denoted as $\mathcal{Y}$ and the cardinality of Y is denoted as $\#\mathcal{Y}$ *or* $|\mathcal{Y}|$.

Vectors and Distances

The distance between two equal length vectors can be expressed through the *Hamming Distance (HD)*

$$\mathbf{HD}(Y; Y') \triangleq \#\{i : Y_i \neq Y_i'\}$$

It may be pertinent to describe the distances as a fraction of the vector instead. In this case, the *Fractional Hamming Distance (FHD)* is used. The FHD is defined thusly:

$$\mathbf{FHD}(Y; Y') \triangleq \frac{\mathbf{HD}(Y; Y')}{|Y|}$$

For definitive purposes, the *Hamming Weight (HW)* may be used, which is described thusly:

$$\mathbf{HW}(Y) \triangleq \#\{I : Y_i \neq 0\}$$

The *Euclidian Distance* between two discrete vectors is defined as:

$$\| Y - Y' \| \triangleq \sum_{i=1}^{|Y|} \sqrt{(Y_i - Y_i')^2}$$

Distributions and Sampling

A discrete random variable has a probability distribution specified by its probability mass function *p(y)*.

$$p(y) \triangleq Pr(Y = y)$$

or by cumulative distribution defined as

$$P(y) \triangleq Pr(Y \leq y)$$

Distribution Parameters

A random variable Y will have an *expectation*, which is the weighted average, or mean of Y and is defined thusly

$$\mathbf{E}[Y] \triangleq \sum_{y \in Y} yp(y)$$

whilst the expectation of of a function of Y, f(Y) is defined as

$$\mathbf{E}_{y \in Y}[f(Y)] \triangleq \sum_{y \in Y} f(y)p(y)$$

The variance of the discrete random variable is defined as

$$\Sigma^2[Y] \triangleq \mathbf{E}_{y \in Y}[(Y - \mathbf{E}[Y])^2]$$

with a standard deviation of

$$\Sigma[Y] \triangleq \sqrt{\Sigma^2[Y]}$$

Statistical Distance

The difference in distribution between the distributions of two discrete random variables Z, and X which consist of samples from the same set $\mathcal{Y}$ can be quantified as a measure of their *Statistical Distance*

$$\mathbf{SD}(A; B) \triangleq \frac{1}{2} \sum_{v \in \mathcal{V}} |Pr(A = v) - Pr(B = v)|$$

Probability Density Functions

Binomial Distribution: The binomial distribution has been presented in a large range of PUF papers, including the seminal work of [2]. Thusly, its relevance of use in PUFs has been established. A binomial distribution describes the number of successes in n independent but identical Bernoulli experiments with success probability p. The probability mass function of the binomial distribution function is given by

$$f_{bino}(t;n,p) \triangleq \binom{n}{t} p^t (1-p)^{n\text{-}t}$$

with a cumulative distribution function

$$F_{bino}(t;n,p) \triangleq \sum_{i=0}^{t} f_{bino}(t;n,p)$$

The expectation of a binomially distributed random variable Y is given by $\mathbf{E}[Y] = np\ \Sigma^2[Y] = np(1-p)$

2.2 Related Work

Although an extensive and broad survey of the existing literature has been conducted, no similar works exist in terms of IoT SAFE. This work will be a first in discussing the use of IoT SAFE in a defence context, or any context at all. With regards to PUFs, there are a number of seminal works to be addressed; the same goes for autonomous, and private 5G networks, including military, private, and private-military.

2.2.1 Physically Uncloneable Functions: Constructions, Properties, and Applications

The primary reference work which addresses PUFs, their differences, and performs a quantitative analysis of various PUFs is the work of Maes "Physically Unclone-able Functions: Constructions, Properties, and Applications" [2].

This work describes the current security landscape, and the need for PUFs, and then goes on to properly, and formally define a PUF. The work is composed as a book, consisting of papers written by the author over a period of three years, as a doctoral dissertation. Especially the chapter "PUF/Based Entity Identification and Authentication" is seen as highly relevant for this thesis. The chapter describes identification as a two/phase process, where for inherent identities, such as PUFS, they have to be enrolled during an *enrolment phase* which consists of collecting the identities of all entities that need to be identified. This phase is followed by an *identification phase* where the purported identity is presented when requested. Maes, determines that inherent identities have some advantages over provisioned identities. Firstly, the identities do not have to be created from a state or a *True Random Number Generator (TRNG)*, since the inherent identity results from the creation process of he entity. Secondly, the identity needs to be stored in some form of non-volatile memory, or some other process which incurs physical changes to the entity. This can, according to Maes, induce a non-negligible cost. Inherent identities, do not need additional storage. Thirdly, Maes argues that reading an identity is generally less intrusive than writing an identity, and can be done faster with a higher degree of reliability. Which Maes argues is of particular interest for entities created in high-volume manufacturing flows, where unit cost is "directly

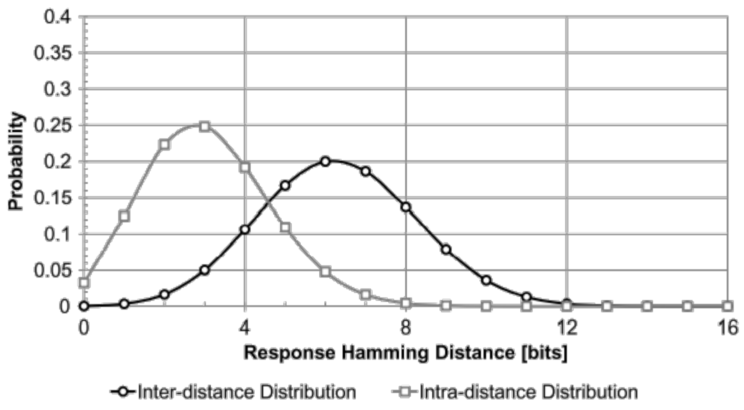


Figure 2.1: Inter-to-inra distance metric

affected by the yield and processing time of each manufacturing step". Finally, Maes describes the downside of the inherent identity, as the manufacturer having no control over what this identity will be.

Further, Maes argues that inherent identifiers are inherently fuzzy, and shows fuzzy behaviour. This means in practice that the behaviour is not entirely uniformly distributed, nor is it perfectly reproducible. Another noteworthy comment is that the fuzzy behaviour has a tendency to become more fuzzy when exposed to noise. Maes argues that in order to make meaningful interpretations of responses from $Eval(puf_i(x)) \in X$ the intra-distance (self-variance) needs to be smaller that the intra distance$Eval(puf_{i,j)(x))} \in X \in PUF_{i,j}$ in this case, one can set a threshold where below the threshold one can assume that the response from the $Eval(x)$ is a intra-distance from the same entity, rather than an inter-distance from another entity. This is what Maes chooses to call the *identification threshold* as seen in figure 2.1. Maes, however neglects to assert that this threshold will require a large amount of data collection on the properties of each individual PUF class, which in turn would need to have inter-distances, that are sufficiently unique to not correlate to another puf_is intra-distance range. For an ideal PUF with a fractional hamming inter-distance of 50% and a fractional hamming intra-distance of 0% would still need to be sufficiently unique.

In practice, this could be done the same way one measures the efficiency and fuzziness problems, already encountered in biometrics. Which are solved by measuring the *False Acceptance, Rejection*, and *Equal Error Rates*. The author proceeds to measure the EER, and plot an ROC curve for all described PUFs, which, as seen in figure 2.2 yields a clear advantage for the ring-oscillator PUF with an EER or 10^{-6}, which is the acceptable minimum for an identification system. The others are at least two orders of magnitude lower. Maes argues that a minimisation of EER to not only acceptable 10^{-6} levels should be achieved, but actually more secure, and critical 10^{-12} should be achieved.

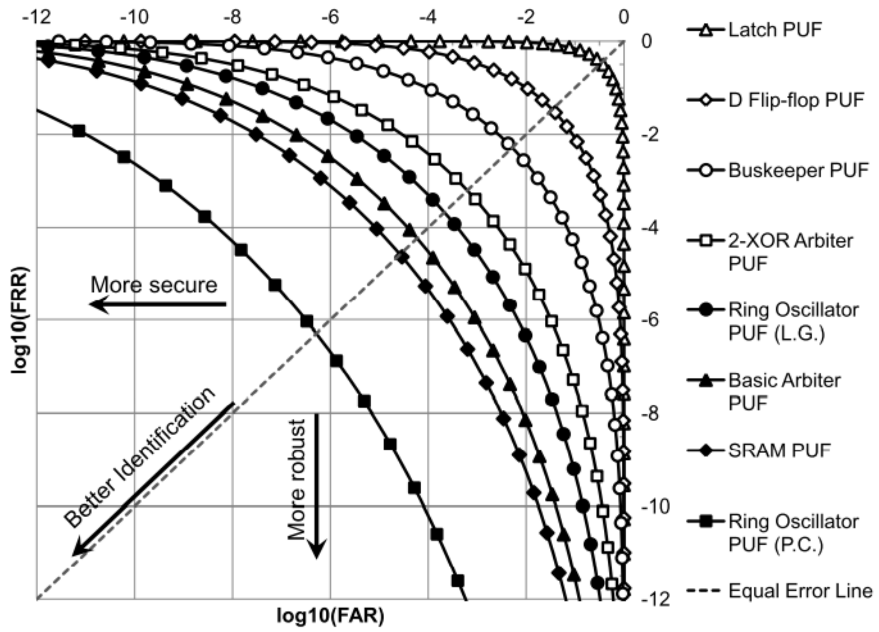


Figure 2.2: ROC for various PUFs

After asserting this, he finds that this can be achieved for all PUF if their size on the silicon chip is increased. The PUF that reaches the 10^{-12} threshold with the smallest silicon area, by far is the SRAM PUF, which Maes also has commercialised in his company "Intrinsic ID", which has a certified and commercially successful implementation of an SRAM PUF called "QuiddiKey". QuiddiKey is essentially an SRAM PUF, which is passed through a en Error Correcting Code (ECC) algorithm for increased reliability, and also a series of hash functions for increased randomness. The final derived key from the SRAM PUF can be provisioned at will and can be used to wrap external keys by encrypting them and storing them as normal data.

2.2.2 IoT Authentication

There are a wast amount of proposed authentication methods proposed for use in IoT based networks, for various purposes [12]. Which one is chosed depends on the needed availability of the network connection, the need for reliability of the connection, as well as the intrinsic need for security which implicitly is determined when choosing an authentication scheme for Iot devices. The communication protocol of the IoT device, will often determine the authentication method used. The communications protocols often used in IoT networks, such as MqTT, 802.11, XMPP, AMQP, Bluetooth and BLE, Cellular, CoAP, DDS, LoRa and LoRaWAN, LWM2M, MQTT, 'Wi-Fi', Zigbee and Z-Wave; all use different authentication methods. For a reliable and simple setup where bandwidth is less of an issue, TCP/IP with various forms of authentication such as WPA2. Although finding a single popular authentication protocol that is designed for IoT hardware authentication, is challenging, research into these is on the rise and more. For the purpose of interoperability and standardisation the authors find that using an already established protocol (DTLS), and adding hardware based authentication to the key exchange process is a much more adaptable approach.

Chapter 3

Methodology

This chapter aims to describe the manner in which this thesis came to be, as well as its products in form of the protocol for the composite key scheme used to authenticate to the TLS session, and serve to secure the SIM card from theft in IoT devices, placed in autonomous military networks. Using the designs from certain commercial applications of the SRAM PUF, as well as general designs of SRAM PUF schemes, the design for this authentication system is based. This chapter firstly describes the method for the production of this thesis in itself, and then goes on to describe the overall goals, both "strategic", and "tactical". Then, the feasibility will be assessed, followed by a risk analysis, and ethical considerations.

3.1 Method

This thesis employs a qualitative method for analysis of literature, and a quantitative method for the design of the authentication system. This thesis uses a literary study approach to evaluate the status of the field of PUFs, especially SRAM PUFs. An evaluation of the field is used in order to determine which approaches, and what technologies are of interest when designing a PUF-based composite authentication system. The literary study composes a fairly large part of this thesis, and will in theory comprise the main portion of this thesis, occupying chapters 1, 2, 3, and parts of 4, and 5.

There are three main ways to approach this problem, depending on the difficulties that may be encountered. One can create a purely theoretical description of the operation of the system proposed, or one could create an implementation that Thales Group Norway Inc can physically see, or one could simply look at the APIs that are within the system. In this case the level of ambition would lead us to at least attempt to implement a combination of a PUF and IoT SAFE so that a device with a IoT SAFE Applet running cannot be authenticated on any other device. In such a case difficulties may be encountered, so it is important to have a strong theoretical description in addition to the implementation in case something were to go wrong. Mainly this thesis will use a qualitative measure in order to determine if the authentication works properly or not, as based on the face that this cannot be qualitatively done in-depth enough. The project will be implemented on a bread-board. If this takes an unreasonable amount of time, then the security issues regarding 5G military networks for IoT will be assessed as part of the thesis. In practice, since the IoT SAFE applet has not yet been created, it will be simulated by a raspberry pi, using mock IoT SAFE API Calls. This can be done, as Thales Group Norway Inc has created a Middleware, which can communicate with higher layers of the architecture, which makes reliance on the IoT SAFE standard itself more abstract https://github.com/ThalesGroup/iot-safe-middleware.

3.2 Literature Review

The literature review is a large and substantial part of this thesis upon which all other ideas, knowledge, and products are based. The databases used were, 'IEEE Xplor', 'Association for Computing Machinery (ACM)', 'NTNU Oria', 'Google Scholar'. The search terms were "("Physically Uncloneable Functions" OR "PUF") AND ("IoT" OR "Internet of Things" OR "Autonomous" OR "Ad-Hoc") AND ("Military" OR "Defence" OR "Defense")" for the main papers, whilst it for survey papers the string AND "Survey" was concatenated. Papers were also found by using references in papers read, including the survey papers. On IEEE Xplor this returned 241,973 results, whilst on ACM it returned 52 results. Google scholar returned a overwhelming 12.000. NTNU returned 1236 results.

3.2.1 Paper Selection

When the searches completed, the pages were looked at and titles were read to assess relevance. The seemingly most relevant papers were opened and then the abstracts were read, and the papers lightly skimmed to further assess their relevance. To assess relevance, it was looked at whether or not the method was novel and useful. If it was not a clone of other approaches, and it indeed was usable, the paper would be read and the method classified, as described in 3.2.2

3.2.2 Classification and Criteria

The papers were classified into two main categories, "more interesting", and "less interesting". This is based on the relevance of the paper to the main goal of the thesis which is authentication using SRAM PUFs. Examples of papers that were classified as "Jess interesting" were papers on other PUF approaches that were still useful enough in their general thought-material to still be regarded. Examples of the papers included in the "more interesting" classification were papers directly related to the surveying of different PUF technologies, and SRAM PUFs in particular.

3.2.3 Paper Selection

The "most interesting" papers were then read, the larger survey papers and books being first, to provide an overview. Subsequently, the more specific papers were read.

3.3 Milestones, Deliverables, and Resources

In order to obtain the necessary knowledge to complete this project, is is a necessity to communicate well with Thales, in order to get a picture of what the current state of the topic is. It is of additional importance to learn how to set up the bread board correctly. As well as reading all the necessary papers in order to actually understand how this all fits together. The project will result in a masters thesis with a theoretical description and description of the practical steps, as well as a physical implementation with proof-of concept code. The deliverables will be aimed to be finished about a month before final delivery of the project, with regular supervision starting at weekly from 1/12/2022 to 1/2/2023, then bi-weekly from 1/2/2023 to 15/4/2023, then reverting to weekly till the project is finished. The project will have a continual dialogue with Thales, throughout the life cycle of the project. When the project officially starts, then the first month is dedicated to doing a literature review, which will continue in a smaller scale alongside the main project. Then, the APIs will be studies and the laboratory environment set up at Gjovik. Following this, and assessing the feasibility the theoretical part is written alongside the experiments, as this is the best option in case you were to forget what you were doing after the experiment is over. This will of course be a continual process. The purpose of this chapter is to convince the reader that you know exactly what to do. This chapter gives a description of how the project is to be broken down into smaller parts and activities.

In order to obtain the necessary knowledge to complete this project, is is a necessity to communicate well with Thales, in order to get a picture of what the current state of the topic is. It is of additional importance to learn how to set up the bread board correctly. As well as reading all the necessary papers in order to actually understand how this all fits together. The project will result in a masters thesis with a theoretical description and description of the practical steps, as well as a physical implementation with proof-of concept code. The deliverable will be aimed to be finished about a month before final delivery of the project, with regular supervision starting at weekly from 1/12/2022 to 1/2/2023, then bi-weekly from 1/2/2023 to 15/4/2023, then reverting to weekly till the project is finished. The project will have a continual dialogue with Thales, throughout the life cycle of the project. When the project officially starts, then the first month is dedicated to doing a literature review, which will continue in a smaller scale alongside the main project. Then, the APIs will be studies and the laboratory environment set up at Gjovik. Following this, and assessing the feasibility the theoretical part is

written alongside the experiments, as this is the best option in case you were to forget what you were doing after the experiment is over. This will of course be a continual process. The purpose of this chapter is to convince the reader that you know exactly what to do. This chapter gives a description of how the project is to be broken down into smaller parts and activities.

3.4 Feasibility Study

Since the author has written on a similar subject for his bachelor thesis, he does not imagine the theoretical part of the task to become excessively complex. It will, however, take some time to learn to do all the wiring and such, but the supervisor has good knowledge in this field. The author also has good programming skills, and can is familiar with working with less-than-well documented APIs. In addition, the fallback plan of a pure theoretical assessment makes this a somewhat safe project. According to my supervisor, what may be difficult is the implementation on the bread-board, where issues may be encountered. In order to mitigate this, it is worthwhile learning how to work with basic electronic circuits as part of the project. It may also be challenging to do the implementation, but at worst this will just take a fair amount of time. Since this is a full-time project, that will not be a major issue. The author expects the PUF and the IoT SAFE standard to not initially work together with our some fiddling, but assesses that the implementation of this project will be possible and yield good results in terms of authentication of a SIM to a singular device.

3.5 Risk Analysis and Contingency Plans

In order to assess the risks in the project is is of importance to first assess the assets of the project. These are: time, skills, communication, and focus. In case of time loss due to the author procrastinating, being sick, spending too much time on a single part etc... It is paramount to consult the supervisor often. In order to mitigate the skills needed, the author needs to set the scope well, and set time off to learn what he does not know. To mitigate communication issues, the author needs to talk to Thales in order to underline the importance of communication for a good quality project. To maintain focus the author needs to have a strict working regiment, and be able to prioritise the thesis, as well as taking good breaks when necessary in order to not get tired of the thesis.

3.6 Ethical and Legal Considerations

The legality of the project is well within bounds, as there are no closed source software that will be used, nor will there be collected any personal data (except for interview notes with supervisor and Thales). The project is not classified, so it can be distributed freely. Regarding ethical considerations, it is positive to protect a nation through strengthening the defence forces. Using autonomous devices instead of people can greatly reduce loss of life in a warfare situation.

Chapter 4

Design and Analysis

This section aims to describe the findings in this paper as they are evaluated in respect to the criteria set in section 3.2.2. The analysis consists of an evaluation of the feasibility of the usage of PUFs as an authentication component, assessing which statistical properties they need to exhibit, in relation to which statistical properties they *do* exhibit. Furthermore a design for such an authentication system is outlined and designed, and a possible implementation is described.

4.1 Statistical Modelling of PUFs

Authentication needs to be repeatable, or else, the identity cannot be repeatedly ascertained. This is one element of the authentication process that PUFs struggle with. The fact is, that PUFs are fuzzy and inconsistent as a general rule. Approximately 10% of the bits in an SRAM puf changes between each measurement, which can be troublesome. Therefore, the trend within PUF-designs is to use Error Correcting Codes (ECC) to preserve a repeatable identity of the PUF instance. An error correcting code scheme can correct up to 25% uncertainty. Which should be good enough for almost all applications. However, there is a need to model the PUFs in order to ascertain which kinds of performance one can expect when using the PUF over a longer period of time, in highly dynamic conditions. This research has been conducted by Maes et. al, along with the claims above relating to the instability or (inter distance) of the PUF instances.

4.1.1 Cost - Resources

PUFs are incredibly cheap to manufacture, and use, in most applications. Since PUFs identities are intrinsic features of a manufactured object, it goes to prove that PUFs require no additional pipeline. Thus, they can be implemented easily and relatively painlessly into most applications which want to take use of their advantages. Either as a PRNG or as an identity source, which should be supported by ECC and key derivation software. For more high security PUF applications, a butterfly PUF can be considered, which is an SRAM PUF that can be reorganised inside of a MOSFET, using VHDL or a similar hardware description language.

4.1.2 Feasibility

A common assumption that is seen when evaluating SRAM PUFs, is that is that spatiality is not of relevance is PUFs, and that the different cells are all independent. This is seen in the works of, among others Maes [2] [3], and [13]. This is not actually the case, as physical closeness of memory cells tends to be en influencing factor to their randomness, especially if they were to be connected. Thus, [14] argues that the entropy estimation of PUFs, should also include this dependency relationship between close memory cells. This can indeed affect local randomness, and the statistical distribution of the PUF as a whole.

When assessing the reliability of PUF-based applications, it can be observed that many require PUF responses to be reliably reproducible, simultaneously as purporting the fact that PUFs are inherently unpredictable [15], [7], [13], [2], [3].

Seeing as PUFs are noisy, and non-uniform, they are often combined with fuzzy extractors [16] [17] [18]. According to to Delvaux et.al, the non-uniformity can be addressed with a more cleverly chosen [n, k] bound when extracting the PUF [19]. Mapping the fuzzy PUF to a reliably to the same key, may be difficult, but using a good fuzzy-extractor, possibly with a similarity-preserving hash function, the process may become more repeatable and reliable. Delvaux et.al also argues that extracting select random strings from the PUF response may aid in the amplification of the privacy when dealing with spatial bit-correlations. These work in such a way that after enrolment and helper-data computation, a uniform bit-string which later can be used to reconstruct the the same cryptographic secret reliably under varying operating conditions, where the helper data may be public. These extractors do have some limitations, as presented by [20], are that they do require a significant amount of physical space, and may also add significant run-time to the process. Therefore Peeters et.al proposed a reverse fuzzy extractor which is purported to be a more lightweight option, where the verifier instead is given the computational burden of the calculations [21] which implies that fresh helper data has to be produced each time, whilst the new PUF response is sent to the verifier for verification. The reverse fuzzy extractor has been criticised for its possibility of opening for a complete reconstruction of the PUF from an adversary, and is in this case also in-viable due to the reverse fuzzy extractor scheme, being out-of-scope for the DTLS authentication process, which is the focus presently.

4.1.3 Randomness of Phase

When assessing the quality of a cryptographic system, the randomness of the sequences are of paramount importance. The randomness of the entire phase or block is important, but also the randomness of a subset of the phase. This can be measured in many different ways, but one of the most prominent ways of doing this is to use the NIST standard for measurement of cartographic randomness

Local Randomness

According to NISTs' "Local randomness: Examples and application" [22], and [23] local randomness is whether a subset of a global sequence is random in an even distribution. If a sequence contained exactly 50% ones and zeroes, the sequence would still not be random if the entire first half was ones, and he second half was zero. Thus, to measure how random the local phenomenon of a sequence are, statistical tests can be employed to this effect. Among these tests are the "diehard tests", the "kendall and smith" tests, and "Persons chi-square test".

In this thesis local randomness, will not be assessed, as this is not the purpose of this thesis, and is considered out of scope.

4.2 Attacks

When creating an authentication system, it is important to keep in mind the attacks one is defending against. The main vectors for attack on PUFs are described in various literature, among these

4.2.1 Emulation attacks

An emulation attack for a PUF is an attack in which an attacker has collected a subset of all Challenge-Response Pairs (CRPs), and from these uses machine learning to predict what the PUF would respond to an arbitrary challenge with a high degree of confidence [**Modelling**]. According to the same authors such an attack could be applied to delay-based PUFs, which an SRAM PUF is not.

4.2.2 side channel attacks

Side channel attacks are attacks which use the inherent physical properties of an IC, such as creating an electromagnetic field, and consuming electrical energy to assess which operations the device is doing, with which data. In theory these techniques could be applied to any device, but there are solutions to these attacks such as de-correlating the processing, adding random overhead to the processing, and otherwise confusing the data. Below follows a further description of some common side-channel attacks

Power Analysis Attacks

A power analysis attack is in its essence a side channel attack that takes advantage of the unintended information leak from various operations performed on an electronic device. The workings of this attack relies on the physical properties of the device, where transistors use different voltages to simulate 1 and 0's, and different input to an arbitrary black box, will produce different power traces based on the input, thus creating an unintended secondary input from which one can infer the input to the black box [24].

A power analysis attack can be divided into a few branches. Firstly, they can be invasive or non-invasive (e.g reading power directly from BUS, or magnetic probe). Secondly, they can be passive or active(listening or providing input), and thirdly, they can be differential or simple (i.e statistical or directly linked [25]).

Simple Power Attacks

Simple Power Analysis (SPA) is a technique for power analysis that will measure the power consumption of a device over a period of time. In a chip, there are different pieces of logic, and sub-chips that are used for different operation, as well as control flow in programming languages (status flags, ALU). This will cause the chip which is performing the operation to use different amounts of power based on the different parts it is using, as these operations and places in the chip require different amounts of power and voltages to operate. A classical example is an if statement. In an if statement the control flow can branch in multiple "directions", based on which route is taken, different amounts of power will be consumed and one can therefore deduce which operation was taken, and one can therefore deduce which action was taken [25] [26]. Conversely, the amount of 1's and 0's read from memory, may also be reflected in the power profile [27], as seem in figure 4.1

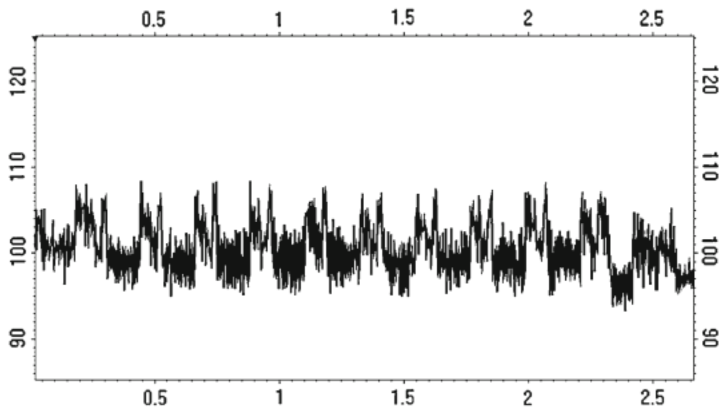


Figure 4.1: AES Powertrace

SPA tends to examine the current state of a circuit as data points plotted to a time axis, thus a "trace" is created. This is possible to do by connecting an oscilloscope with a sufficient resolution, to the power-bus. From this power trace, the operations conducted can be deduced. An example would be this AES powertrace, where the "rounds" of encryption are clearly visible.

Differential Power Analysis Attacks

In contrast to SPA, Differential Power Analysis utilises statistical means to analyse the power consumption of a circuit. In DPA multiple traces are made, and they are then subsequently compared to each other. When the power traces are compared, statistical trends may appear where the correlated power traces differ. Thus, given a set of power traces, one can determine commonalities where there is substantial noise as only correlated sets give trace excursions. In effect, this means that with a large number of traces that are made by giving random input to the device, one can infer minute correlations as non-correlated trace extrusions do not appear on the correlational graph. This effectively removes noise in the data set [25] [26] [17].

4.2.3 fault injection attacks

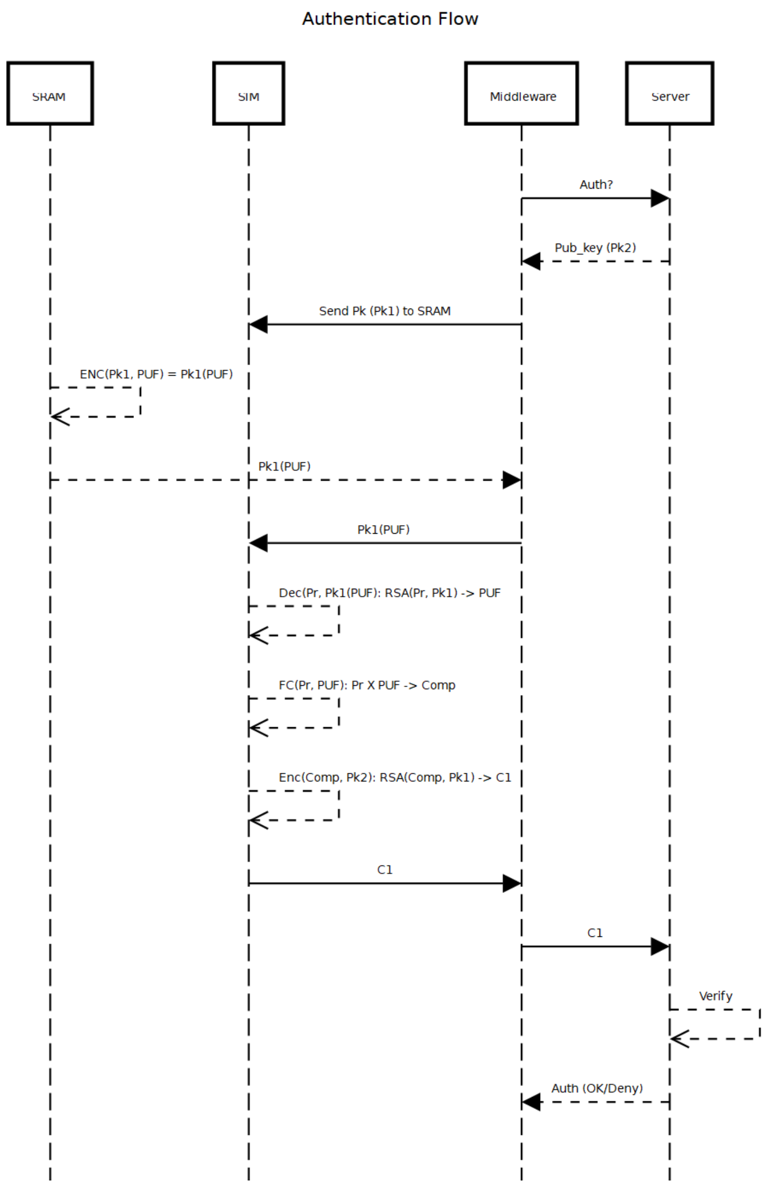
A fault injection attack is when the attacker injects a bit-error, electrically or otherwise in order to disrupt the processing of data in a chip, in order to create a different state in the program such as an error state in order to reach some part of the program which would normally be inaccessible. An example of this is injecting a fault at the correct time in a program, flipping a specific bit from zero to one, and thus perhaps changing the path of an if statement. This can be circumvented by programming differently, and is not an intrinsic property of the chip at hand. This may or may not be applicable to a PUF based authentication system.

4.2.4 Implementation Design

The implementation is designed to be a composite key-approach, where the key from the PUF is used in composition with the stored key on the IoT SAFE SIM, to be used as the private key for RSA key exchange in a TLS-session. The components involved are the PUF itself (and its associated processing), the IoT SAFE SIM, the IoT SAFE Middleware, and the conversing party to which the PUF-holding device is the supplicant. The goal of this authentication system is to make sure that the SIM cannot be used with any other devices, than the one it is "paired" with. In the case of e.g. a mobile phone, this would mean that the phone with the SIM, could authenticate to the military 5G slice with an authenticated TLS-session, but removing the SIM and using it in another device, would not work.

To accomplish this the device and the SIM must somehow be linked, and the authenticator must only accept a key that is a composite of the two units. The authentication flow is the following. The device initiates an authentication session, by means of the middleware, which tasks the SIM to send a SYN. The server then sends its public key to the middleware. The middleware tasks the SIM to send its public key to the SRAM module, where the derived key from the PUF is enciphered with the SIMs public key. The middleware then then sends this this enciphered derived key to the SIM. The SIM then deciphers the PUF derived-key inside its secure hardware, and then subsequently signs the PUF-derived key with its private key. This key is then enciphered in the servers public key, and sent back. From this the server will only accept the PUF if it is signed with the private key of the IoT SAFE sim.

To illustrate this flow, figure 1 is attached:


Authentication Flow
SRAM
SIM
Middleware
Server
Auth?
Pub_key (Pk2)
Send Pk (Pk1) to SRAM
ENC(Pk1, PUF) = Pk1(PUF)
Pk1(PUF)
Pk1(PUF)
Dec(Pr, Pk1(PUF): RSA(Pr, Pk1) -> PUF
FC(Pr, PUF): Pr X PUF -> Comp
Enc(Comp, Pk2): RSA(Comp, Pk1) -> C1
C1
C1
Verify
Auth (OK/Deny)

The PUF itself does need a some processing to be able to be useful and effective. First and foremost, one needs to realise a couple of things. Firstly, PUFs do not have perfect intra-distance, meaning that there is some fuzzyness to the PUF. Secondly, using the PUF key directly, is not a good idea as this does not facilitate multiple users, nor is super-safe. Therefore, the PUF needs to be error-corrected, and also derived keys need to be created from this main "physically-derived-PUF-key". An example of this is the system implemented by Intrinsic ID, where the SRAM data is read, and helper data is created from the fuzzy PUF-source, which is precludes subsequent fuzzy extraction upon a re-extraction of the physical-PUF-key. This fuzzy extractor, will use ECC with the helper data, and subsequently hash the output. This hashed output is combined with the helper data to create a variable for a Key Derivation Function (KDF). It one of the keys from this KDF which will be sent back to the SIM and be used to create the RSA key for (D)TLS authentication.

From a physical standpoint, this implementation can be implemented on any device that has an IoT SAFE SIM, processing power, and an SRAM module (or MOSFET in which an SRAM module could be programmed). An SRAM Module can be bought cheaply from various online sources, but it does need a fair amount of memory to have a large enough key-space. According to Intrinsic ID intrinsic the there is a need for at least 1kb of ram for a 256 bit key. When implementing an SRAM solution, there are many considerations to take into account. Among these are, The aforementioned bit-error-rates, silicon ageing, environmental conditions, and consistency. When implementing a PUF using anti ageing, ECC, and the fact that temperatures are not exerting a majorly significant instability, the SRAM PUF works well when implemented as a normal SRAM module with some post-processing, as described in the introduction to this chapter.

4.3 Experimental Setup

This section describes how the experiment is set up, and how the different measurements are taken. Firstly the physical setup is seen schematically and physically, and then the software is explained, followed by a subsection on how the different measurements for were taken

4.3.1 Physical Setup

The experimental setup consists of a hardware part, and a software part. The hardware part of the physical design, is implemented on a bread-board, with Arduino Nano, and Uno micro-controllers to read-write, and process, and send data, to and from a computer. The SRAM module is connected to the Arduino Nano, using the SPI interface, and is read using serial-mode, from byte 0x00 to 64000, as the SRAM module is 512Kb (23LC512-I/P-ND IC SRAM 512KBIT SPI/QUAD 8DIP). These bytes, are then stored in a 64B buffer, and then sent to an Arduino Uno as bytes, which then subsequently prints the bytes sent as using the binary print parameter in the Arduino print function. This binary data is then stored in a file, fed through an ECC in the form of digital-lockers, then subsequently used as key-material input to a key-derivation function in the form of PBKDF2.

The experimental setup schematic is shown in figure 4.2 which illustrates an SRAM Chip connected to an Arduino Nano, which in turn is connected to an Arduino Uno, which is connected to a computer. The SRAM chip is communicates with the Arduino Nano using the SPI protocol, which is realised through a connection from the SRAM SO/SI (Slave Out/Slave In) ports which connect to the MOSI and MISO ports on the Nano respectively. The /CS is active low, and is connected to the digital output port on the Nano, so it can switch between multiple SRAM chips if needed, but is pulled low with a 1k pull down-resistor, to avoid an inadvertent high. The opposite is true for /HOLD which is pulled high through a 1k pull up-resistor to avoid an inadvertent low. The poles for Vcc and GND are bridged with a 0.1 μF capacitor to reduce electrical noise. PIN 6 on the SRAM is the clock input from the Nano. The Nano and the Uno share a common ground to avoid a short-circuit, an then the Tx of the Nano is connected to the Rx of the Uno. Since only one Serial signal can be sent at once, the data is needed to be read and sent in bursts.

The figure seen in 4.3.1 shows the physical setup of the hardware components, that run the software described above. The SRAM chip can be seen on the top of the image on the left on the breadboard, the Arduino nano is on the right of the breadboard, whilst the Uno is on the In this setup a series of measurements were taken from an SRAM chip, and then this chip was exchanged for another chip, five different SRAM chips were measured.

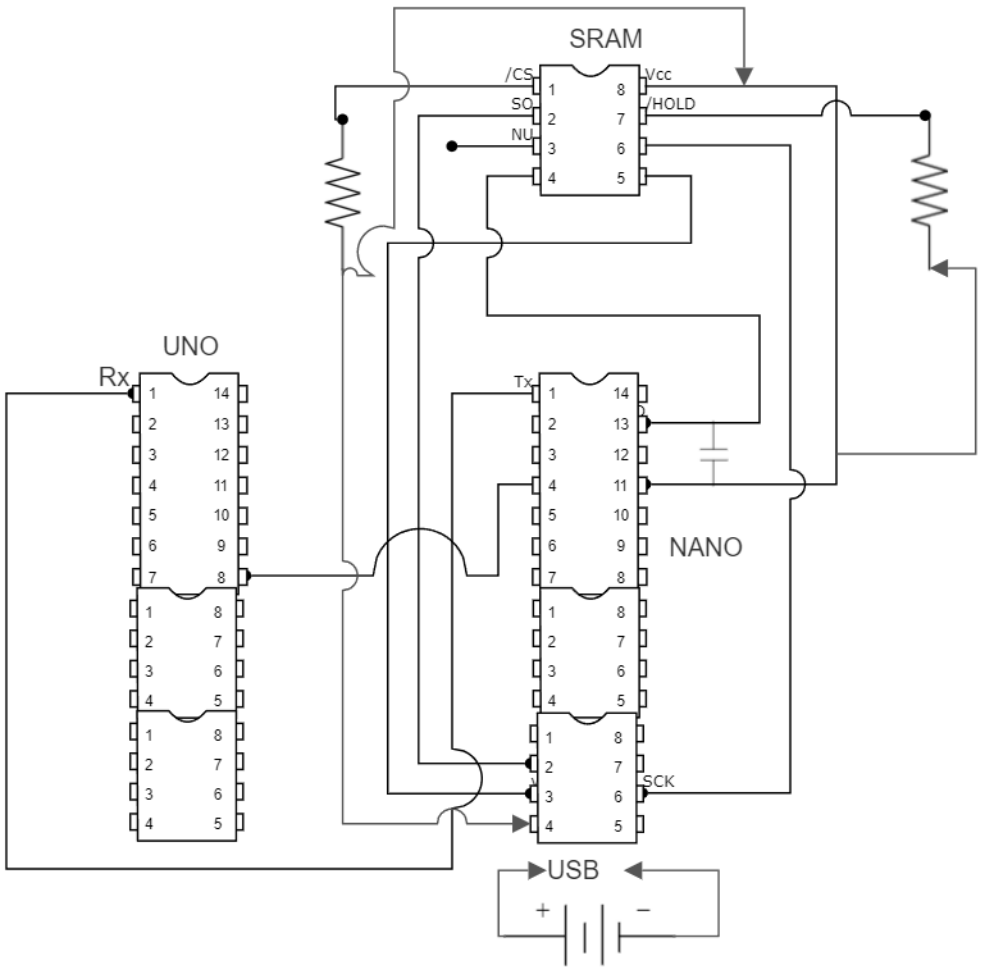


Figure 4.2: Setup Schematic

4.3.2 Software Setup

The software for the NANO is shown in figure 4.3.2. This software sets the integer constants needed to communicate with the SRAM chip, as defined in its data sheet, then sets up the Arduino for serial sequential communication. Following this, the chip select is set to active, the chip is set to read mode, and the serial read from the SRAM begins, and is then transmitted to the python program via the Serial.print command, which sends bit-converted bytes to the python program, which then writes these bytes to a file for further analysis and processing.

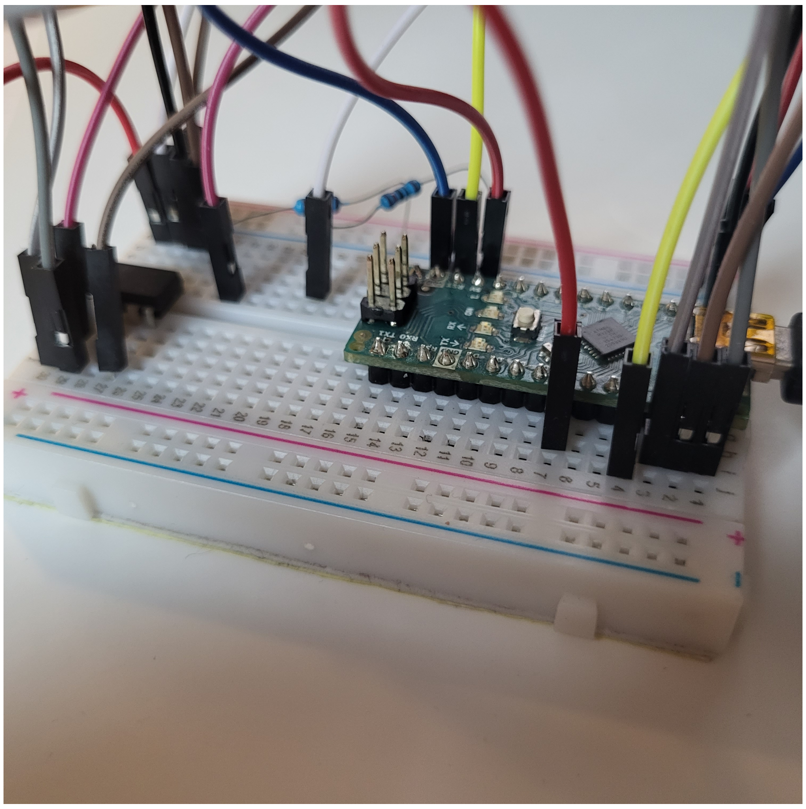


Figure 4.3: Physical Setup

Code listing 4.1: Arduino Nano Read Sequential SRAM with SPI

```
#include <SPI.h>
#include <SpiRAM.h>

#define RDMR       5
#define WRMR       1
#define READ       3
#define WRITE      2
#define RSTIO      0xFF
#define ByteMode   0x00
#define Sequential 0x40
#define CS         10
#define RST        9

static uint32_t address = 0;
const long long numBits = 512000;
const long long numBytes = 64000;
SpiRAM SpiRam(0, CS);
void _set_mode(char mode);
void enable();
void disable();
char _current_mode;

void setup(void) {
  Serial.begin(115200);
  SPI.beginTransaction(SPISettings(20000000, MSBFIRST, SPI_MODE0));
}

void loop() {

  delay(10000);
  int i;
  int address = 0;
  _set_mode(STREAM_MODE);
  enable();
  SPI.transfer(READ);
  SPI.transfer((char)(address >> 8));
  SPI.transfer((char)address);
        for (i = 0; i <= 64000; i++)    {
    Serial.print(SPI.transfer(0xFF), BIN);
        }
  Serial.print("\nin␣file!");
  Serial.print("\ndonedonedone");
  delay(2000);
  disable();
}

void enable()
{
  digitalWrite(CS, LOW);
}

void disable()
{
  digitalWrite(CS, HIGH);
}

// Mode handling
void _set_mode(char mode)
{
  if (mode != _current_mode)
  {
    enable();
    SPI.transfer(WRSR);
    SPI.transfer(mode);
    disable();
    _current_mode = mode;
  }
}
```

The code that constructs the secret PUF-key is seen in figure **??**. This code reads the binary PUF-data from a file, and creates the helper data if it does not already exist. If it does exist the file is simply opened. Then, the fuzzy extractor library uses digtal locker-coding to extract a reliable PUF from the binary data, which is then hashed, and written to a file. This hash is then sent to the key-derivation function so that the master key will have a length which is compatible with PBKDF2.

Code listing 4.2: Bit extractor implementation

```
import os
import serial
import time
from pathlib import Path
import os
import hashlib
from fuzzy_extractor import FuzzyExtractor
from Crypto.Protocol.KDF import PBKDF2
from Crypto.Hash import SHA512
from Crypto.Random import get_random_bytes
import ssdeep
from Crypto.Cipher import AES
from Crypto.Hash import SHA256
from Crypto.PublicKey import ECC
from Crypto.Signature import DSS
import pickle


def read_bits():
    while True:
        try:
            with serial.Serial('/dev/ttyUSB0', 115200, timeout=30) as conn:
                print(conn.name)
                with open(f"raw_sram_bytes{int(time.time())}", "wb+") as f:
                    f.write(conn.read_until("donedonedone"))
                    print("read␣until␣done␣token")
        except Exception:
            pass

    return True

def generate_key():
    with open("./raw_sram_bytes1685186332", "r") as f:
        sram_raw = f.read()

    helperfile = Path('helper.txt')
    sha3hash = hashlib.sha3_512()

    extractor = FuzzyExtractor(215326, 8)
    if (os.path.exists(helperfile) == False):
        with open(helperfile, "xb") as f:
            key, helper = extractor.generate(sram_raw)
            print(helper)
            print(key)
            pickle.dump(helper, f)
    else:
        with open(helperfile, 'rb') as f:
            helper = pickle.load(f)
            print('loaded␣helperfile')
            key = extractor.reproduce(sram_raw, helper)
            print('reproduced␣key')

    sha3hash.update(key)
    hashed_key = sha3hash.digest()

    with open("hashed.key", "wb+") as keyfile:
        keyfile.write(hashed_key)

    password = hashed_key
    salt = get_random_bytes(16)
    keys = PBKDF2(password, salt, 64, count=1000000, hmac_hash_module=SHA512)
    key1 = keys[:32]
    print(key1)

    return key1





def main():
    if read_bits():
            SRAM_key = generate_key()
    else:
        print('Could␣not␣read␣bits␣from␣PUF.␣Exiting...')
        exit()

if __name__=='__main__':
    main()
```

The following program receives the key that the enrolment program has created, and does the mock authentication 4.3.2

Code listing 4.3: Bit extractor implementation

```
from Crypto.Protocol.KDF import PBKDF2
from Crypto.Random import get_random_bytes
from Crypto.Cipher import AES
from Crypto.Hash import SHA256
from Crypto.PublicKey import ECC
from Crypto.Signature import DSS

def mock_SIM(SIM_AES_key):
    file_in = open("encrypted.bin", "rb")
    nonce, tag, ciphertext = [ file_in.read(x) for x in (16, 16, -1) ]
    file_in.close()

    cipher = AES.new(SIM_AES_key, AES.MODE_GCM, nonce)
    decrypted_key = cipher.decrypt_and_verify(ciphertext, tag)

    # Emulating server key
    ECDSA_key = ECC.generate("ed25519", get_random_bytes)
    h = SHA256.new(decrypted_key)
    signer = DSS.new(ECDSA_key, 'fips-186-3')
    signature = signer.sign(h)

    return signature
```

```
def main():
    SRAM_key = get_key_from_file()
    SIM_AES_key = get_random_bytes(16)
    cipher = AES.new(SIM_AES_key, AES.MODE_GCM)
    encrypted_SRAM_key, tag = cipher.encrypt_and_digest(SRAM_key)
    nonce = cipher.nonce
    file_out = open("encrypted.bin", "wb")
    [ file_out.write(x) for x in (nonce, tag, encrypted_SRAM_key) ]
    file_out.close()

    signed_key = mock_SIM(SIM_AES_key)
    print(signed_key)


if __name__=='__main__':
    main()
```

4.3.3 Measurements Taken

Since it was not possible to acquire a real IoT SAFE SIM, so as to actually test the IoT SAFE in physical hardware, it was decided to simulate the functions provided from the SIM in software. As this thesis focuses to more on the applicability of using a PUF as a secondary input to a composite key, and to look at the authentication rates of this system, rather than looking at the physical aspect of the hardware communication, such as latency this is acceptable. There is, however, need for hardware in order to get real PUFs from the SRAM, which is why the physical setup was created.

This means that the hardware PUF generates a key from its bit-states, which then will be sent to a key extraction and generation program
Each SRAM chip, was powered on, read, powered off, and reread 300 times, this amount was chosen as a result of time constraints as turning the chip on and off practically requires a reboot of the Arduino. the results are then compared for inter and intra distance metrics, randomness, and used to assess the effectiveness of the key-creation in terms of a how reproducible it is.

Power Draw

From the technical specifications of the SRAM chips used in this Thesis, their voltage is 5.5V, and their current is 3mA. In sum this is 0.00001374999 Watt/hours per three seconds, which is the amount of time it takes to read 64000 bytes from the SRAM chip at 115200 baud. Overall this is negligible. Comparing this to a battery pack of one AAA battery at 1.87 mAH, this would equal 136000 hours of operation, which roughly equates to 1.5 years of operation if under constant load. This is seen as a time frame longer than a standard military deployment of an average 6-12 months, on a single charge. According to Thales "According to a 2011 article, a typical Canadian soldier may carry a set of fifteen AA batteries and two CR123 batteries upon exiting a forward operating base. If a mission is expected to last more than three hours, a further set of batteries will be carried. Once the expected mission time exceeds 24 hours, a soldier will carry a third set of AA and CR123 batteries – a total of 51 batteries"[28].

Chapter 5

Results

5.1 Intra-Distance

Measuring the hamming distance between the start-up reads of the SRAM chip, these were compared with a program that read all the files and compared them all to the first read of the chip, which was set as a baseline. The following python script was used to this effect.

the script takes the first measurement as a base, to which it compares the succeeding files. If the files for some reason are of different lengths, the end of the longer file is appended to the shorter file, producing a hamming distance of 0 for the padding. The measurements and their averages are then written to a file

Code listing 5.1: Hamming distance measurement

```
import os
import distance
from distance import levenshtein

def padd_base(diff_len, file, base):
    print(f'difference␣in␣length␣is:␣{diff_len}')
    padding = file[-diff_len]
    padded_base = base.append(padding)
    return padded_base

def padd_instance(diff_len, file, base):
    print(f'difference␣in␣length␣is:␣{diff_len}')
    padding = base[-diff_len]
    padded_file = base.append(padding)
    return padded_file

def main():
    directory = f'{os.getcwd()}/compare_dir'
    filenames = []
    hamming = []
    levenshtein = []
    count = 0
    total = 0

    for path in os.listdir(directory):
        if os.path.isfile(os.path.join(directory, path)):
            count += 1

    for i in range(count-1):
        filenames.append(f'{directory}/to_compare_{i}')

    with open(filenames[1], 'rb+') as base:
        base = base.read()
        print(f'basefile:␣{filenames[0]}')
        for file in filenames[1:]:
            with open(file, 'rb+') as instance:
                print(f'read␣instance␣file:␣{file}')
                instance = instance.read()
                file_difference = len(instance) - len(base)
                if file_difference > 0:
                    print("instance␣too␣large,␣padding␣base")
                    temp_file = padd_base(file_difference, file, base)
                elif file_difference < 0:
                    print("instance␣too␣small,␣padding␣base")
                    temp_base = padd_instance(file_difference, file, base)
                else:
                    temp_base = base
                    temp_file = file
                try:
                    print('Calculating␣hamming␣distance...')
                    ham_dist = distance.hamming(temp_base, temp_file)
                    print('calculated!')
                    hamming.append(ham_dist)
                    print(f'Instantial␣hamming␣distance␣from␣base:␣{ham_dist}')
                except Exception:
                    print('Oops')
                    pass


    print(f'␣the␣Hamming␣distance␣is␣{hamming}')
    for result in range(len(hamming)):
        total += hamming[result]
    average = total / len(hamming)
    print(f'average␣hamming␣distance␣is␣{average}')

    print('writing␣raw␣data␣to␣file')
    with open('raw_data', 'a') as f:
        f.write(f'hamming␣distances:\n')
        for measurement in hamming:
            f.write(f'{hamming}\n')
        f.write(f'average␣hamming:␣{average}')



if __name__=="__main__":
    main()
```

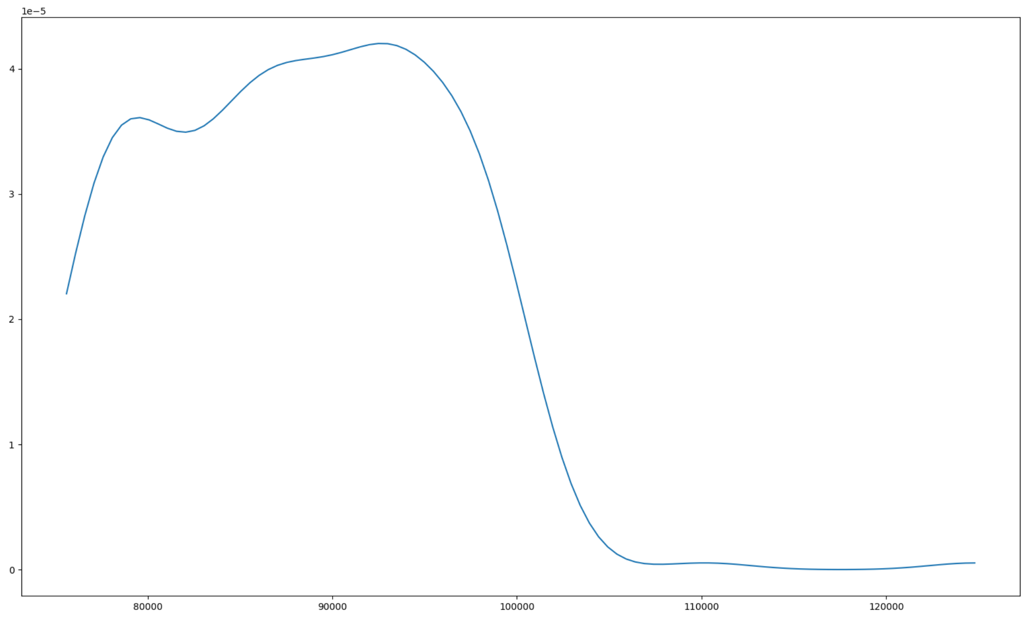


Figure 5.1: SRAMCHIP 1

This script produced the values seen in the figure below, where the values are plotted as values in a probability density function, and the average of all these was 88296.950 which equates to 17.25% of 512000. This measurement Shows a distribution of values in the range of 75602 to 124797. This gives a distribution of likelihood for intra-distance in the range of 14.77% to 24.8%, where the most extreme outlier of 124797 hamming-distance occurred exactly once, with a probability of this occurrence being very low, as can be seen in the binomial distribution 5.6. The graphs 5.1 5.2 5.3 5.4 5.5 show the intra distance measurements for all the 5 SRAM chips measured, which all lay in the same range.

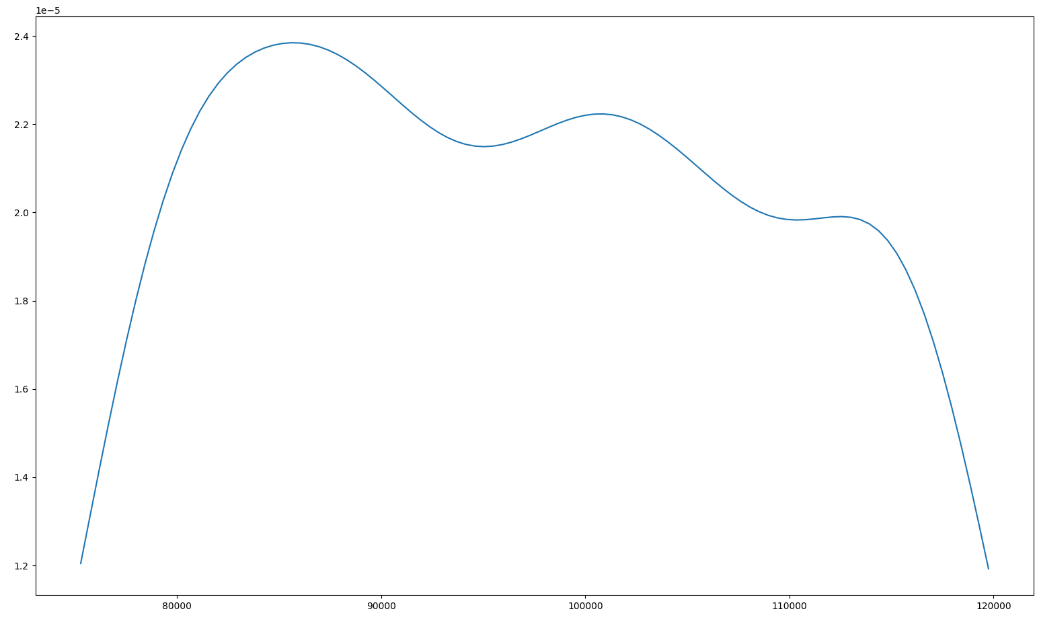


Figure 5.2: SRAM CHIP 2

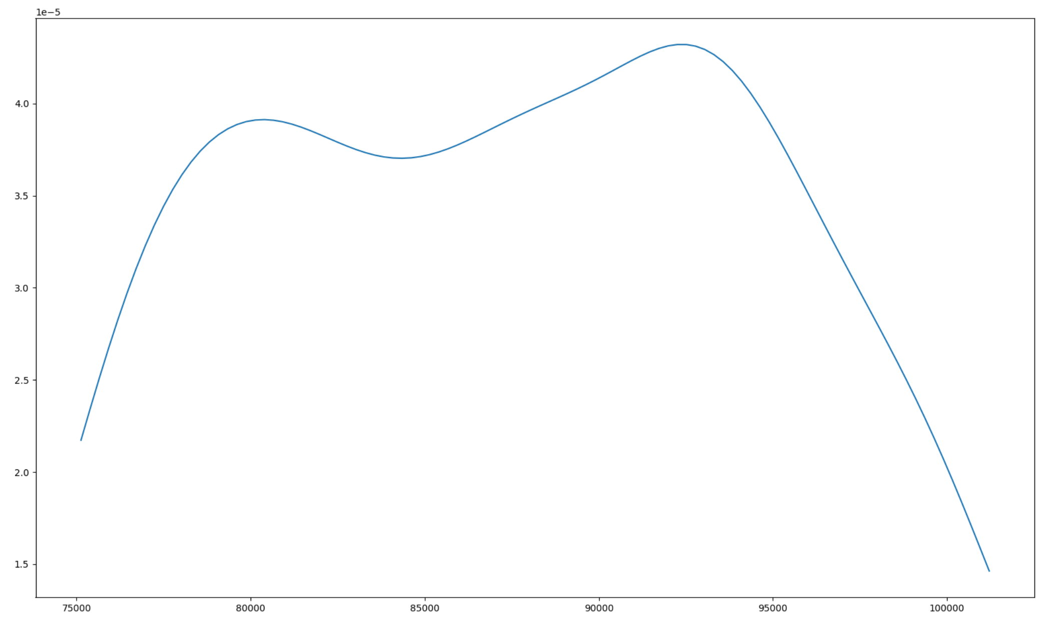


Figure 5.3: SRAM CHIP 3

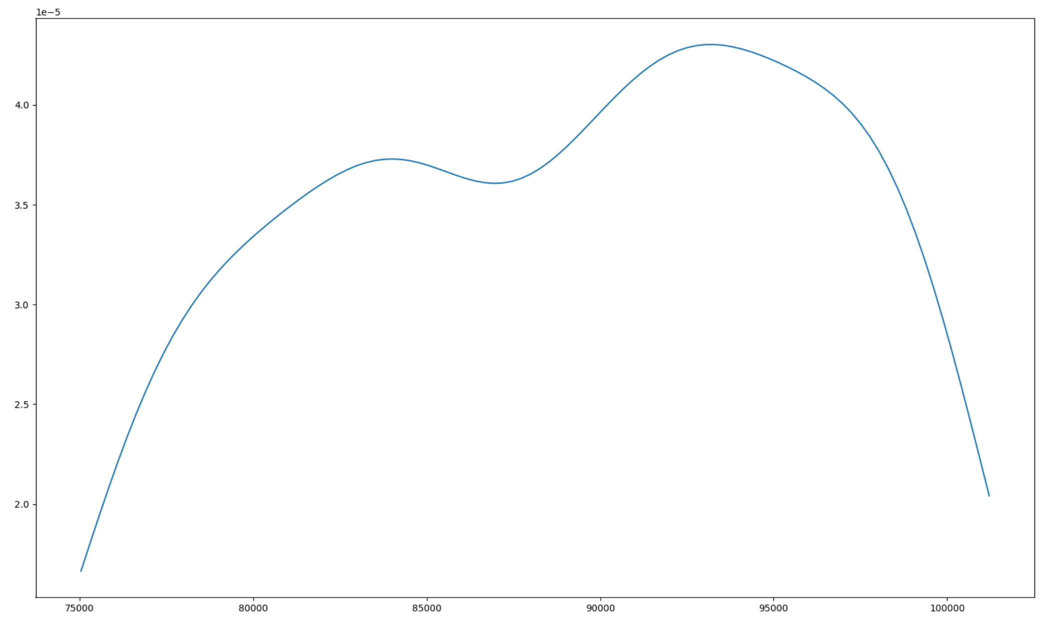


Figure 5.4: SRAM CHIP 4

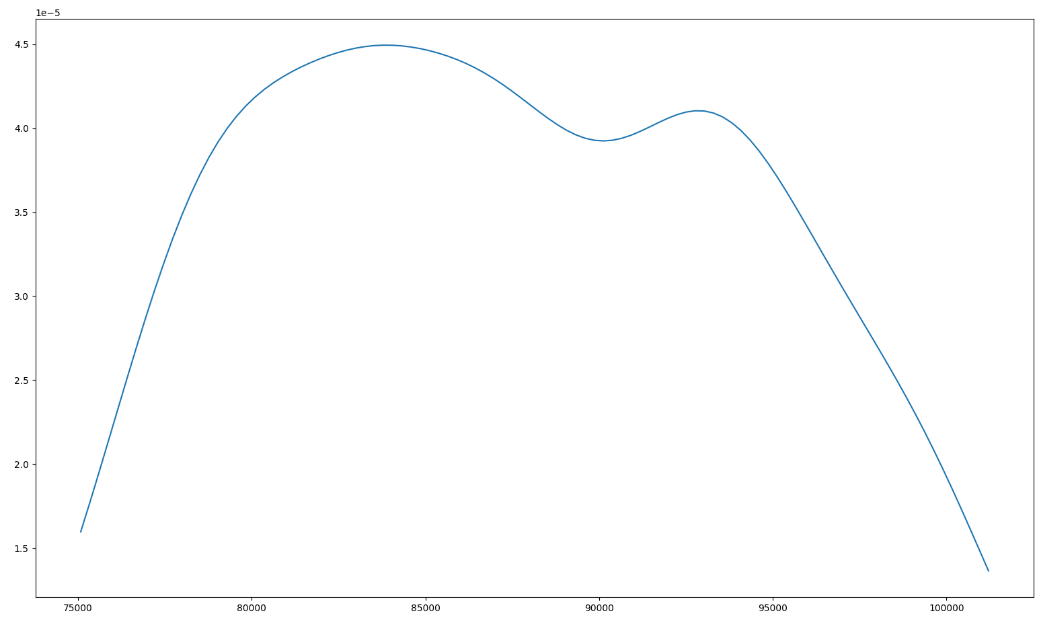


Figure 5.5: SRAM CHIP 5

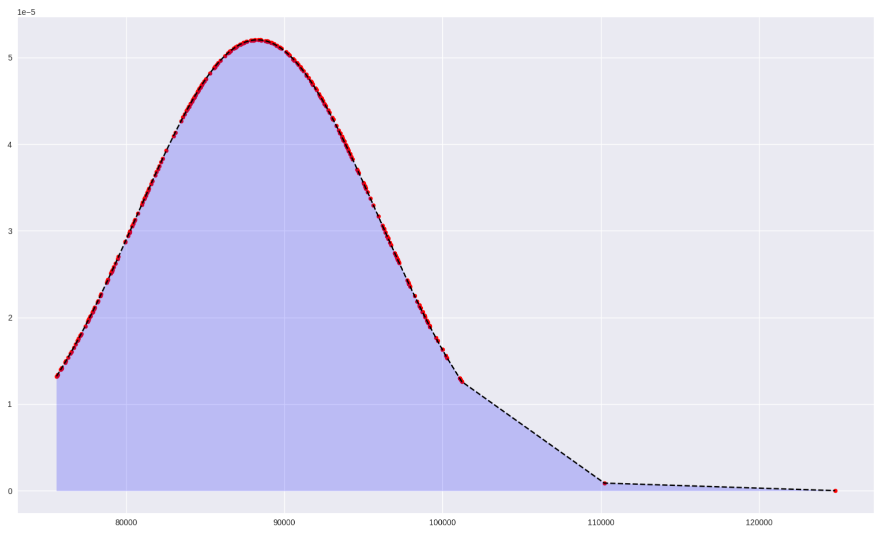


Figure 5.6: common binomial distribution

5.2 Inter-Distance

Between the SRAM chips, is was observed a significant hamming distance as measured between the ICs. The hamming distance between the SRAM chips measured the same way as for the inter-distance-metric, where the responses from the SRAM chips are compared by hamming-distance from each other giving 25 measurements, from where each SRAM chip is compared with the output of all other SRAM chips. This can be seen in the table below. where the average of all these is 256296 which equates to 50.06% inter-distance. Changing the keys given to the mock SIM resulted in a non non-match, which is to be expected for a non-mated sample, which increases the confidence in the observed inter-distance [29].

SRAM #	1	2	3	4	5
1	0	256680	257368	257344	253880
2	256680	0	255392	256744	256784
3	257368	255392	0	256920	256448
5	257344	256744	256920	0	255400
5	253880	256784	256448	255400	0

5.3 Key Generation with PUF

as can be seen in the code-listing 4.3.2 the extracted PUF is used to generate a key from the fuzzy-extracted PUF, which was corrected with the helper-data. The key is then hashed. This program was ran for all the PUFs which were read and did not ever fail to generate the same SHA3 hash for the same PUF. This indicates that the system works as intended. Whether the authentication system will work as well in a practical setting is hard to determine, and according to [2] [30] The PUG instance evaluation may be affected by external factors, such as the temperature, voltage level etc. Although this is the case, other research specifies that the PUFs change minimally with changes in temperature, within such a range that it is not within a range with a need for correction [3] [31].

Chapter 6

Discussion

This chapter aims to ground the previous chapters, and give them a critical outlook in terms of performance, experimental conditions, and realism. This chapter will also answer to how the experimental results compare to the expected results, as well as what this can indicate about PUFs for use in authentication systems.

6.1 Discussion of Viability

As discussed in the analysis chapter 4 the viability of the system relies on the reliability of PUF-responses which will be used for the key generation. As such, the viability of the system is currently in such a state that the inter-distance metrics of a PUF will be within the tolerance limit for an authentication system, as is. In the current state, without error correcting codes in the raw PUF-responded, the results were mediocre. With the error correcting codes applied, as used in the key generation scheme, the authentication scheme is yet to fail. The values produced by the error-corrector would be interesting to measure, but this has not been done in this thesis. This is due to the fact there are only five RAM-chips, and this is hardly enough data to make any sort of usable conclusion about the effectiveness of the fuzzy-extraction.

Besides the fact that no usable conclusions about fuzzy-extraction can be made, the helper data size from the fuzzy-extractor implementation used, digital lockers, which produces a substantial amount of helper data. It produces so much helper data, in fact, that it becomes unusable with larger (i.e. realistically sized) SRAM fingerprints. Just for this project, the helper data size was 43 GB! This in effect also makes the program unbearably slow, having to calculate and write such (relatively) large amounts of data. According to [32]

According to U.S. Department of Energy Office of Scientific and Technical Information (OSTI) [33] the SRAM PUF provides a small input space for key-generation, which can be used for cryptographic key-generation, or as a seed for Pseudo-Random Number Generator(PRNG), but can be noisy, and can therefore often

need a fuzzy extractor. This point exacerbates the point that PUFs can be useful, but do need correction to work properly. Currently a method for correction which uses a reasonable amount of storage is not publicly available, although improvements to [16] have been proposed with a claimed improvement of 98% in terms of required storage space [32].
The PUFs do work, however, and so does the authentication system. The degree to which it is usable is unknown. Using a second-factor creating an intrinsic link between the hardware and a SIM-card definitely has potential. There is, however, a question which arises in this case. This being whether an eSIM will obsolete the need for a PUF-based hardware root of trust, as they cannot be moved between devices anyway. If this is possible is unknown. One would still have to store the IoT-SAFE keys somewhere, so why not utilise a PUF to this effect.

6.2 Evaluation of Design

When assessing the design choices made in this thesis, it is important to keep in mind that the IoT SAFE standard is not finished, nor publicly available, and thus may change. The design in this thesis may still be applicable irrespective of this. Creating a hardware root of trust which is near-impossible to replicate is an interesting design choice for lightweight, portable applications, where storage, space, voltage, and security is of importance. Such an application is for military use. The design aims to be reasonably secure, but is not formally proved as in the formalisation of PUFs as in [5], this is a weakness of this thesis.

Chapter 7

Conclusion

7.1 Conclusion

The definition of a PUF itself remains subject to academic debate, as scholars hold differing views on the precise characteristics and criteria necessary for classifying a PUF. The determination of what constitutes a "Weak" or "Strong" PUF is also an area of disagreement within the academic community.

There are additional extensions to the concept of PUFs that enhance their functionality and capabilities, namely the re configurable PUF and the public PUF. The rPUF is a device capable of intentionally changing its response to the same input challenge.

For more high security PUF applications, a butterfly PUF can be considered, which is an SRAM PUF that can be reorganised inside of a MOSFET, using VHDL or a similar hardware description language.

The main vectors for attack on PUFs are described in various literature, among these are emulation attacks which are emulation attacks in which an attacker has collected a subset of all Challenge-Response Pairs, and from these uses machine learning to predict what the PUF would respond to an arbitrary challenge with a high degree of confidence.

The design of a PUF authentication system for IoT-SAFE enabled devices communicating using the DTLS protocol, have been prototyped and shown to be able to generate repeatable keys with an acceptable level of security. SRAM PUFs have also been applied practically, and have proven themselves to have beyond minimum EER, of up to 10^{-12}. SRAM PUFs have proven themselves to use low power, be repeatable, and implementable.

Research questions posed in 1, they are presented once more, and answered.

- What are the theoretical applications and limitations of using PUFs for autonomous systems?

The theoretical applications of using PUFs, especially SRAM PUFs for authentication in autonomous systems are their intrinsic usefulness of taking up little physical space, and providing a strong hardware root of trust that which intrinsically mitigates a wide variety of threats. Some threats are intrinsic to all hardware based systems, but can to a degree be countered by modern measures such as secure programming, hardware shielding, and secure storage. What makes PUFs especially useful is that they can also be used as a seed for random number generators, which in secure communication is an important feature.

The limitations of the SRAM PUF is currently the digital space taken up by the helper data, and the time required to calculate this. Besides this, the ram-chips have an appropriate amount of space, and even an integrated SRAM IC of 2kbit and less is enough for an at least 256 bit key, which is practical. The fact that there are no public libraries available to create and interact with PUFs is certainly a limitation of the technology in the current stage, which is halting research. An observation was made on various internet-pages, which indicated that a fair amount of researchers and interested parties struggled with the practical implementation of PUFs, due to the low-level skills needed to interact with such a system, which is often not attributed to the average security researcher, at least in the theoretical dimension.

- Will the use of a PUF significantly improve the ease an security of autonomous military systems, from the theft of SIM-Cards with IoT SAFE?

Being able to intrinsically link the communication key to a device is a useful property of the authentication system designed in this thesis. if one were to do a non-mated comparison with a SIM card which has been enrolled with a server, then one would most probably get a non-matched result, which is the goal of the authentication system. SRAM PUFs have, as mentioned, been shown to have a low EER. The theft of SIM cards may still happen, but the authentication would most likely not work. This property also applies in reverse, where a device without a sim-card, also cannot authenticate. Thus, stealing a SIM-card, or a device will yield the same result of a non-match.

- How reliable are the methods presented for use in authentication, in terms of statistical repeatability, and changing environmental conditions? Including power consumption.

The reliability of the methods presented is seemingly acceptable, in the sense of repeatability, as shown by authors of numerous papers, and the intra and inter distance metrics measured in this thesis. In terms of changing environmental conditions, it seems from various authors that temperature change has a negligible effect on SRAM PUFs, within reasonable ranges of -125 to 125 deg C. The most prominent factor which causes reliability issues is ageing. Ageing can to a degree be mitigated, quite profoundly by applying anti-ageing techniques.

Even in cases where the full anti-ageing approach may not be feasible, there are still strategies that outperform doing nothing. A significant practical advantage of the proposed anti-ageing solutions is that they can be implemented without making any circuit modifications or requiring pre-deployment efforts. Therefore, they can be easily incorporated into standard SRAM implementations within a typical development process. It is important to note that zeroing SRAM PUFs (for security purposes) with a fixed pattern, such as all zeros, has a detrimental effect on each of the PUF's quality measures, and is strongly discouraged. Overwriting the PUF response with randomly generated data in real-time is a preferable alternative.

7.2 Future Work

In the future the author would like to see a full authentication system be created using SRAM PUFs, where there is a formal definition of the authentication system in use, so that the customer can have a strong foundation when considering using such a technology. The author would also like to see FRR, FAR, and EER metrics of this system be presented so that there is ground evidence for its dependability as for any fuzzy authentication system.

The creation of public tools and libraries is also an are that should be developed to create a wider availability of SRAM-PUFs, both for commercial and academic use, which includes anti-ageing, fuzzy-extraction, and pre-defined key-extraction functions, as this is well-described and quite ready for a standard adaptation.

7.3 Acknowledgements

The author would like to thank Stephen Wolthusen for guidance during this thesis. The author would also like to thank Thales for proposing this thesis, and giving valuable feedback during its production.

Bibliography

[1] 5.-V. Consortium. '5g vinni: Vertical innovation infrastructure.' (2022), [Online]. Available: https://www.5g-vinni.eu/ (visited on 10/02/2022).

[2] R. Maes, 'Physically unclonable functions: Constructions, properties and applications,' 2013.

[3] A. J. Menezes, S. A. Vanstone and P. C. V. Oorschot, *Handbook of Applied Cryptography*, 1st. USA: CRC Press, Inc., 1996, ISBN: 0849385237.

[4] M. G. Kendall and B. B. Smith, 'Randomness and Random Sampling Numbers,' *Journal of the Royal Statistical Society*, vol. 101, no. 1, pp. 147–166, Dec. 2018, ISSN: 0952-8385. DOI: 10.1111/j.2397-2335.1938.tb01115.x. eprint: https://academic.oup.com/jrsssa/article-pdf/101/1/147/49708753/jrsssa_101_1_147.pdf. [Online]. Available: https://doi.org/10.1111/j.2397-2335.1938.tb01115.x.

[5] F. Armknecht, R. Maes, A.-R. Sadeghi, F.-X. Standaert and C. Wachsmann, 'A formalization of the security features of physical functions,' in *2011 IEEE Symposium on Security and Privacy*, 2011, pp. 397–412. DOI: 10.1109/SP.2011.10.

[6] GSMA. 'Iot safe: Iot sim applet for secure end-to-end communication.' (2021), [Online]. Available: https://www.gsma.com/iot/iot-safe/#doc (visited on 01/06/2021).

[7] N. M. Karie, N. M. Sahri, W. Yang, C. Valli and V. R. Kebande, 'A review of security standards and frameworks for iot-based smart environments,' *IEEE Access*, vol. 9, pp. 121 975–121 995, 2021. DOI: 10.1109/ACCESS.2021.3109886.

[8] B. Insider. 'We asked executives about the internet of things and their answers reveal that security remains a huge concern.' (2015), [Online]. Available: https://www.businessinsider.in/We-Asked-Executives-About-The-Internet-Of-Things-And-Their-Answers-Reveal-That-Security-Remains-A-Huge-Concern/articleshow/45959921.cms (visited on 21/01/2015).

[9] GSMA. (2021), [Online]. Available: https://www.gsma.com/iot/wp-content/uploads/2021/06/IoT-SAFE-Whitepaper-2021.pdf (visited on 06/2021).

[10] GSMA. (2019), [Online]. Available: https://www.gsma.com/iot/wp-content/uploads/2019/12/IoT.04-v1-Common-Implementation-Guide-1.pdf (visited on 12/2019).

[11] W. Che, F. Saqib and J. Plusquellic, 'Puf-based authentication,' in *2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2015, pp. 337–344. DOI: 10.1109/ICCAD.2015.7372589.

[12] M. El-hajj, A. Fadlallah, M. Chamoun and A. Serhrouchni, 'A survey of internet of things (iot) authentication schemes,' *Sensors*, vol. 19, no. 5, 2019, ISSN: 1424-8220. DOI: 10.3390/s19051141. [Online]. Available: https://www.mdpi.com/1424-8220/19/5/1141.

[13] K. K. Landes, 'A scrutiny of the abstract,' *Bulletin of the American Association of Petroleum Geologists*, vol. 35, no. 7, p. 1660, 1951.

[14] C. Wachsmann and A.-R. Sadeghi, 'Physically unclonable functions (pufs): Applications, models, and future directions,' in *Physically Unclonable Functions*, 2014.

[15] J. F. Claerbout, 'A scrutiny of the introduction,' *The Leading Edge*, vol. 10, no. 1, pp. 39–41, 1991.

[16] Y. Wen and Y. Lao, 'Efficient fuzzy extractor implementations for puf based authentication,' in *2017 12th International Conference on Malicious and Unwanted Software (MALWARE)*, 2017, pp. 119–125. DOI: 10.1109/MALWARE.2017.8323964.

[17] S. Chen, B. Li and C. Zhou, 'Fpga implementation of sram pufs based cryptographically secure pseudo-random number generator,' *Microprocessors and Microsystems*, vol. 59, pp. 57–68, 2018, ISSN: 0141-9331. DOI: https://doi.org/10.1016/j.micpro.2018.02.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0141933116303441.

[18] M. Adeli, N. Bagheri, H. Martín and P. Peris-Lopez, 'Challenging the security of "a puf-based hardware mutual authentication protocol",' *Journal of Parallel and Distributed Computing*, vol. 169, pp. 199–210, 2022, ISSN: 0743-7315. DOI: https://doi.org/10.1016/j.jpdc.2022.06.018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0743731522001538.

[19] J. Delvaux, D. Gu, I. Verbauwhede, M. Hiller and M.-D. (Yu, 'Efficient fuzzy extraction of puf-induced secrets: Theory and applications,' in *CHES*, Springer, 2016, pp. 412–431. DOI: 10.1007/978-3-662-53140-2_20.

[20] K. Yasunaga and K. Yuzawa, 'On the limitations of computational fuzzy extractors,' 2018.

[21] R. Maes, R. Peeters, A. Herrewege, C. Wachsmann, S. Katzenbeisser, A.-R. Sadeghi and I. Verbauwhede, 'Reverse fuzzy extractors: Enabling lightweight mutual authentication for puf-enabled rfids,' Jan. 2012, pp. 374–389, ISBN: 978-3-642-32945-6. DOI: 10.1007/978-3-642-32946-3_27.

[22] H. Fu and C. A. Miller, 'Local randomness: Examples and application,' *Phys. Rev. A*, vol. 97, p. 032 324, 3 Mar. 2018. DOI: 10.1103/PhysRevA.97.032324. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.97.032324.

[23] M. G. Kendall and B. B. Smith, 'Randomness and Random Sampling Numbers,' *Journal of the Royal Statistical Society*, vol. 101, no. 1, pp. 147–166, Dec. 2018, ISSN: 0952-8385. DOI: 10.1111/j.2397-2335.1938.tb01115.x. eprint: https://academic.oup.com/jrsssa/article-pdf/101/1/147/49708753/jrsssa_101_1_147.pdf. [Online]. Available: https://doi.org/10.1111/j.2397-2335.1938.tb01115.x.

[24] T. K. R. Spreitzer V. Moonsamy and S. Mangard, 'Systematic classification of side-channel attacks: A case study for mobile devices,' 2018, pp. 465–488.

[25] e. a. Kocher. P, 'Introduction to differential power analysis attacks,' May 2011, pp. 5–27.

[26] e. Kocher. P, 'Differential power analysis,' 1999, pp. 388–397.

[27] C. Z. OFlynn. C, 'Chipwhisperer: An open-source platform for hardware embedded security research,' 2015, pp. 388–397.

[28] Thales, 'Puf modeling attacks: An introduction and overview,' in *Reducing the battery burden on the dismounted soldier*, 2016.

[29] F. Farha, H. Ning, K. Ali, L. Chen and C. Nugent, 'Sram-puf-based entities authentication scheme for resource-constrained iot devices,' *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5904–5913, 2021. DOI: 10.1109/JIOT.2020.3032518.

[30] S. Kerr, M. S. Kirkpatrick and E. Bertino, 'Pear: A hardware based protocol authentication system,' in *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS*, ser. SPRINGL '10, San Jose, California: Association for Computing Machinery, 2010, pp. 18–25, ISBN: 9781450304351. DOI: 10.1145/1868470.1868476. [Online]. Available: https://doi.org/10.1145/1868470.1868476.

[31] M. A. Gurabi, O. Alfandi, A. Bochem and D. Hogrefe, 'Hardware based two-factor user authentication for the internet of things,' in *2018 14th International Wireless Communications Mobile Computing Conference (IWCMC)*, 2018, pp. 1081–1086. DOI: 10.1109/IWCMC.2018.8450397.

[32] J. Cheon, J. Jeong, D. Kim and J. Lee, 'A reusable fuzzy extractor with practical storage size: Modifying canetti et al.'s construction,' in Jun. 2018, pp. 28–44, ISBN: 978-3-319-93637-6. DOI: 10.1007/978-3-319-93638-3_3.

[33] T. Bauer and J. Hamlet, 'Physical unclonable functions: A primer,' *IEEE Security amp; Privacy*, vol. 12, no. 06, pp. 97–101, Nov. 2014, ISSN: 1558-4046. DOI: 10.1109/MSP.2014.123.

Appendix A

Additional Material