Chapter 1
Introduction and Preview

1.1 Introduction

1.1.1 Trust and Security in a Modern World

Trust is a sociological concept expressing the positive belief that a person or a system we interact with will behave as expected. In our day-to-day life, we constantly and often implicitly put our trust in other parties, e.g.:

- When we drive a car, we trust that the car will function as expected, that the brakes will work and that the car goes right when we turn the steering wheel right. We also trust that the other people driving cars around us are qualified to drive a car and are paying attention to traffic.
- When we deposit our money in a bank account, we trust that the bank will keep the money safe.
- When we send someone a letter, we trust the postal services to deliver the letter in a timely manner to the right person, and to keep the letter closed such that no one else can read its content.
- When we buy something in a shop, we trust the shop owner to deliver the product, e.g. when we pay in advance, and that we receive the genuine product we paid for. On the other hand, the shop owner trusts that we will pay for all products we carry out.

In the majority of situations, such trust-based interactions work out in the right way, because the parties we interact with are *trustworthy*. In fact, our entire complex society is based on such trust relations between people and systems, and it would not last very long if no one or no thing could be trusted.

However, we don't live in an ideal world, and it would be very naive to think that everyone is intrinsically trustworthy. Many parties have external motives to behave in a trustworthy manner, e.g. the shop and the bank won't get many customers when they cannot be trusted, and the other car owners will primarily drive carefully for their own safety. Some parties cannot be trusted at all; we immediately think of criminals and terrorists, but this can also include e.g., disgruntled employees, envious colleagues or nosy neighbors, or even normally honest people who are tempted to abuse a situation when it presents itself. We need systems that induce, guarantee or even enforce trustworthiness of parties in our non-ideal world. This is what we call *security*, i.e. security is a means to enable trust.

In the past, and to a large extent still today, security is either based on physical protection and prevention measures, on observation and detection of untrusted elements, or on legal and other reprimands of trust violations, and often on a combination of these techniques. For example, in order to keep its (your) money secure, a bank will store it in a vault (physical protection). The access to this vault is moreover strictly limited to the bank's employees and protocols are in place to keep other people away (detection). Finally, by law, trying to rob a bank is also a criminal act for which one will be prosecuted if caught (legal reprimands). In our rapidly digitalizing modern world, these security techniques are by themselves often no longer sufficient to adequately enable trusted interactions, both due to (i) the nature of these interactions, and (ii) the scale of the possible threats.

(i) The remote and generic nature of many digital interactions lacks physical protection and assurance measures, many of which are even implicitly present in non-digital communications. For example, in the past, most interactions with your bank would take place inside the bank's building, face-to-face with one of the bank's employees. You (implicitly) trusted the authenticity of this interaction, e.g. because the building was always in the same place, and perhaps because you physically recognized the clerk from previous transactions, and vice versa. However, in the last couple of years, interactions with your bank have shifted largely to online banking systems. In such an online system, e.g. a website, this implied notion of authenticity no longer exists, since anyone could set up a website resembling that of your bank, and even fake its web address. The same holds from the bank's perspective: anyone could log in to the website and claim to be you. Other security measures are needed to guarantee the authenticity of this interaction.

(ii) The main success of digitalization is that it enables automation of information processes to very large scales and speeds. However, this is also one of the main risk factors when it comes to digital crime. For example, in the real (non-digital) world, there is a risk of having your wallet stolen on the street. However, a thief will have to focus on one victim at a time, and for each attempt there exists a significant risk of failure which often ends in getting caught. In a vastly interconnected computer network like the Internet, with hundreds of millions of simultaneously active users, a *digital* thief can deploy a computer program which targets thousands or millions at a time at an incredibly fast pace. Moreover, failed attacks typically go by unnoticed or are hard to trace back, and even with a very small success rate the thief will get a significant return due to the vast number of targeted victims. Like the threat, the security measures will also need to be digitized and automated in order to offer adequate protection.

1.1.2 Information Security and Cryptology

Information Security

Information security deals with securing interactions involving the communication of information. The need for information security has existed for a long time, historically in particular for matters of love and hate, i.e. secret love letters and sensitive warfare communication. However, in the last couple of decades this need has risen exponentially due to our vast and ever increasing reliance on digital information processing and communication systems. Unimaginable quantities of private and often sensitive information are stored and communicated over the Internet and other digital networks every second. Through the pervasiveness of smart mobile personal devices, digital technology impacts our daily lives in ways we could not have foreseen, and with the introduction and tremendous success of social networks, it has even become an integral part of our lives. In many ways our society has become a flow of digital information, and reliable information security techniques are indispensable to enable trust in this digital world.

Information security techniques are most comprehensibly classified by means of the goals they aim to achieve. The most important goals are:

- *Data confidentiality* relates to keeping information secret from unauthorized parties, e.g. when accessing your bank account statements online, you don't want anyone else to see this information.
- *Entity authentication* deals with obtaining proof of the identity and the presence of the entity one is interacting with, e.g. in an online banking system, you need proof that you're dealing with the real website of your bank, and your bank needs proof that you are who you claim to be before granting access to your account.
- *Data integrity and authentication* is aimed at preventing and detecting unauthorized alteration of data (integrity) and ensuring the origin of the data (authentication), e.g. when you issue an online bank transfer, your bank needs to be sure that it was you who issued the transfer, and that the data of the transfer (amount, account number, ...) has not been changed by someone who could have intercepted the transfer message before it reached the bank.

Cryptology

Cryptography, a subfield of cryptology, deals with the construction of protocols and algorithms to achieve information security goals, typically on a mathematical basis. The other subfield of cryptology is cryptanalysis, which analyzes the security of cryptographic constructions by attempting to *break* their anticipated security. Both subfields are intimately linked, often exercised by the same persons, and a close interplay between both is invaluable. One of the, if not *the* basic principle of modern cryptology is the understanding that a cryptographic construction can only be considered secure if its internal workings are general knowledge and have successfully withstood elaborate cryptanalysis attempts from independent parties. This is also called *Kerckhoffs' principle* after Auguste Kerckhoffs who first stated it [66], and stands in contrast to so-called *security-through-obscurity* which attempts to reach security goals with undisclosed and hence unanalyzed constructions.

A basic design principle for many cryptographic constructions is to reduce the security goal they attempt to achieve to the secrecy of a single parameter in the construction, called the *key*. The obtained level of security is typically expressed by the required effort to break it without knowing the key, which should be an exponential function of the key's length in bits. An important aspect in these security reductions is the assumptions one makes about the power of the adversary, e.g. whether he can observe a number of inputs and/or outputs of a primitive and whether he is only a passive observer or if he can actively or even adaptively interfere with the execution of the primitive. Based on the nature of a reduction, different security notions can be distinguished:

- *Heuristic security* means that even after elaborate cryptanalysis of a construction, no attacks can be found which break its security with a computational effort less than expressed by the key length.
- *Provable security* means that, through logical reasoning, the construction's security can be shown to be equivalent to a mathematical problem which is perceived to be *hard*, with the problem's hardness expressed by the key length. Examples of such hard mathematical problems for which no efficient algorithms are known, and which are actually used in cryptographic constructions, are factorization of large integers (e.g. as used in the RSA algorithm [111]) and computation of discrete logarithms (e.g. as used in the Diffie-Hellman key exchange protocol [31]).
- *Information-theoretical security* means that it can be shown through information-theoretical reasoning that an adversary does not have sufficient information to break the construction's security. This basically means the construction is unbreakable, even to an adversary with unlimited computational capabilities.

For an extensive overview of the construction and properties of cryptographic primitives, we refer to [96]. Cryptographic primitives can be classified based on the nature of their key. We respectively distinguish (i) unkeyed primitives, (ii) symmetric-key primitives, and (iii) public-key primitives and list some of their most important instantiations and achieved security goals.

(i) Unkeyed primitives are constructions which do not require a key. Following Kerckhoffs' principle, their operation is hence completely public and can be executed by everyone. Their security is basically grounded in the difficulty of finding an input which matches a given output. The most used unkeyed primitives are cryptographic hash functions, which provide data integrity and also often serve as a building block in larger cryptographic constructions.

(ii) Symmetric-key primitives are based on a single key which is only known to authorized parties and secret to anyone else. Symmetric-key encryption algorithms, such as block ciphers and stream ciphers, provide data confidentiality between parties knowing the secret key. Symmetric-key message authentication codes provide data integrity and authentication and entity authentication between parties knowing the key.

(iii) Public-key primitives are based on a key pair, one of which is public and the other is kept private. In a public-key encryption scheme, everyone can encrypt a message with the public key, but only the party which knows the private key can decrypt it. In a public-key signature scheme, only the party knowing the private key can generate a signature on a message, and everyone can use the public key to verify that party's signature. Signature schemes provide entity authentication, among other goals.

1.1.3 Physical Security and Roots of Trust

Physical Security

To use a cryptographic primitive in practice, it needs to be implemented on a digital platform in an efficient manner. Unlike Kerckhoffs' principle for the general construction, for the implementation it is typically assumed that the primitive behaves like a *black box*, i.e. one is only able to observe the input-output behavior of the implementation, not its internal operations. In particular, for nearly all (keyed) cryptographic primitives, it is assumed that:

- A secure (random, unique, unpredictable, . . .) key can be generated for every instantiation of the primitive. This is called *secure key generation*.
- The key can be assigned to, stored and retrieved by the instantiation without being revealed. This is called *secure key storage*.
- The instantiation can execute the cryptographic algorithm without revealing any (partial) information about the key or about internal results, and without an outsider being able to influence the internal execution in any possible way. This is called *secure execution*.

While these are convenient assumptions for mathematical security reductions, from a practical perspective they are very hard to attain. Moreover, it is clear that none of these three black-box assumptions can be achieved through information security techniques, but require physical security measures. In a way, one could say that cryptographic primitives reduce information security objectives into physical security requirements.

The fact that none of the three identified physical security objectives are trivial is made clear by the numerous cases where information security systems are attacked by breaking the security at a physical level.

- The fact that secure key generation is difficult was just recently made clear again by Lenstra et al. [77], who show that there is a significant shortage in randomness in a large collected set of actually used public keys from a public key signature scheme, likely caused by badly implemented key generators. For some of the keys in the analyzed collection, this leads to an immediate loss of security.

- Storing secret keys in a highly secure manner partially contradicts the fact that they still need to be in some (permanent) digital format to be usable in an algorithm. For typical digital implementations, this means that the key bits reside somewhere in a non-volatile digital memory on a silicon chip. Even with extensive countermeasures in place, it is very difficult to stop a well-equipped and/or determined adversary from gaining physical access to key memories, e.g. as demonstrated by Torrance and James [143] and Tarnovsky [138].
- There are many ways an adversary can break the secure execution assumption, both on the software and on the hardware level. Modern cryptographic implementations can no longer ignore side-channel attacks, which abuse the fact that all actions on a digital platform leak information about their execution through so-called side channels, e.g. through their execution time [68], their power consumption [69], their electro-magnetic radiation [107], etc. Fault attacks [10] on the other hand seek to disrupt the expected execution of a cryptographic algorithm through physical means, and learn sensitive information from the faulty results.

Physical Roots of Trust

In order to provide these physical security objectives, we cannot rely on mathematical reductions anymore. Instead, we need to develop physical techniques and primitives which, based on physical reasoning, can be *trusted* to withstand certain physical attacks and can hence provide certain physical security objectives. We call such primitives *physical roots of trust*. Figure 1.1 shows how information security objectives can be achieved from physical security and eventually from physical roots of trust, i.e. trusted primitives which are rooted in the actual physical world. Possible candidates of physical roots of trust are:

- True random number generators or TRNGs [37, 122] harvest random numbers from truly physical sources of randomness and can therefore be trusted to produce highly random keys for cryptographic purposes.
- Design styles for digital silicon circuits have been developed which minimize and ideally eliminate certain physical side channels [141].
- Physically unclonable functions or PUFs produce unpredictable and instance-specific values and can be used to provide physically secure key generation and storage. They are the main subject of this book.

1.2 Preview

1.2.1 Introducing Physically Unclonable Functions

A physically unclonable function or PUF is best described as "an expression of an inherent and unclonable instance-specific feature of a physical object", and as such

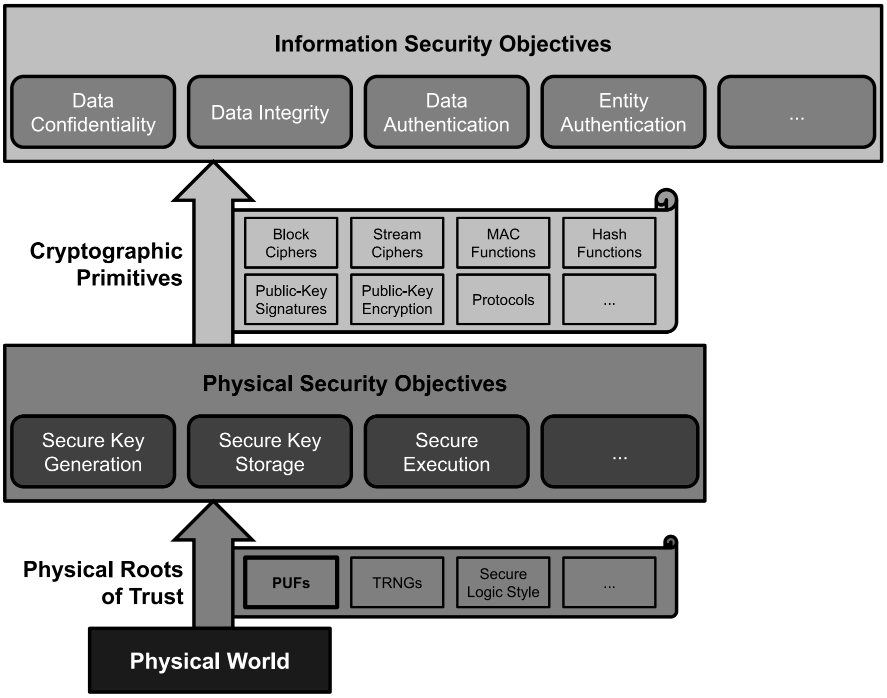

Fig. 1.1 Relations between information security, cryptography, physical security and physical roots of trust

has a strong resemblance to biometric features of human beings, like fingerprints. To be specific, PUFs show qualities which cannot be obtained from cryptographic reductions, but require a physical basis to establish them, the most noteworthy being *physical unclonability*. This means that through physical reasoning it is shown that producing a physical clone of a PUF is extremely hard or impossible.

PUF Constructions The physical motivation for claiming unclonability of an inherent instance-specific feature is found in the technical limitations of the production of physical objects. Even with extreme control over a manufacturing process, no two physically exactly identical objects can be created due to the influence of random and uncontrollable effects. Typically, these influences are very small and only take effect at (sub-)microscopic scales, but leave their random marks nonetheless. A high-precision measurement of these marks serves as an inherent and instance-specific feature. Moreover, creating a second object which produces a similar measurement is infeasible from a physical perspective, and often even technically impossible. Generating such a measurement with an accuracy high enough to distinguish these instance-specific features is the primary goal in the study of *PUF constructions*. The basic technique which is typically used is to design a construction, either external or internal to the object, which amplifies these microscopic differences to practically observable levels.

PUF Properties A wide variety of PUF constructions based on this principle are possible and have been proposed, considering objects from many different materials and technologies, each with their own specific intricacies and useful properties. In order to apply PUFs to reach physical security objectives, a generic and objective description of these *PUF properties* is required. Moreover, it is important to distinguish truly PUF-specific properties from other useful qualities which are inherent to specific constructions but cannot be generalized to all PUFs.

PUF Applications Based on their unclonability and other useful properties, PUFs can fulfill a number of physical security objectives when applied in the right way. Besides taking advantage of the physical security properties of PUFs, such *PUF-based applications* also need to deal with the practical limitations of the construction. This is accomplished by deploying a PUF in a scheme together with other primitives that enhance its qualities. Deploying a PUF in a larger system typically leads to trade-offs, and hence optimization problems, between the aspired security level and the implementation restrictions of the application. Based on an analysis of such a scheme, some PUF constructions will offer better trade-offs than others.

1.2.2 Book Outline

In this book, we study PUF constructions, properties as well as applications, both from a conceptual and from a very practical perspective. Figure 1.2 shows how these subjects relate to each other and are organized in this text.

In Chap. 2, we explain the details of the PUF concept and provide an extensive overview of existing PUF constructions with a focus on so-called *intrinsic* PUFs. This overview is of an unprecedented completeness and serves as a great aid in understanding the true nature of what we rather intuitively have called a PUF. Based on this overview, we also manage to identify significant subclasses, design techniques, implementation properties and even open problems related to PUF constructions.

In Chap. 3, we identify and define different meaningful properties attributed to PUFs and, based on Chap. 2, we analyze if and to what extent actual PUF constructions attain them. From this analysis, a number of these properties are found to be defining for the concept of a PUF, while others are mere convenient qualities but are in no way guaranteed for all PUFs. In order to increase their potential in theoretical constructions, a highly formal framework for using PUFs and their most important properties is also discussed.

In Chap. 4, we discuss the implementation of a significant subset of studied intrinsic PUF constructions on a realistic silicon platform (65 nm CMOS ASIC) and experimentally verify their behavior at nominal condition and at extreme temperature and voltage corners. We capture the qualities of each studied construction in

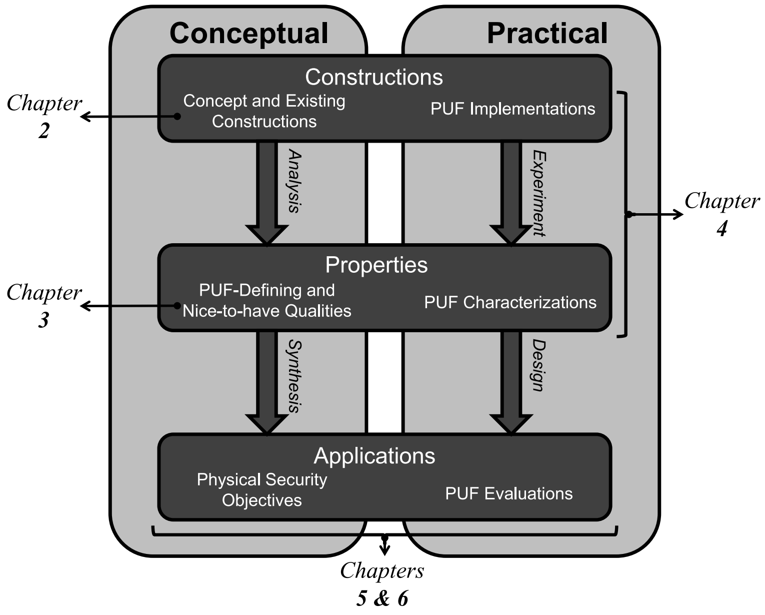

Fig. 1.2 Organization of the subjects in this book and its chapters

a small number of meaningful statistics. Additionally, we analyze the unpredictability of each PUF by introducing heuristic upper bounds on their entropy density.

In Chap. 5, we investigate how PUFs can be used to identify distinct objects, and ultimately provide entity authentication. Quality metrics for assessing identification performance are discussed and applied on the PUF constructions studied in Chap. 4, yielding a classification of their identifying capabilities. We discuss a PUF-based authentication protocol innovatively combining a PUF and other primitives. Authentication performance metrics similar to those for identification are assessed for the constructions from Chap. 4.

In Chap. 6, it is explained how PUFs can be used to obtain secure key generation and storage. Existing notions and techniques for key generation are discussed, and a practical new variant is proposed which yields a significant gain in efficiency. Based on the design constraints of a convenient construction of a practical PUF-based key generator, the PUF implementations from Chap. 4 are assessed for their key generation capacities. To conclude, we present a front-to-back PUF-based key generator design and a fully functional FPGA reference implementation thereof.

In Chap. 7, we summarize the most important aspects of this book and propose a number of interesting future research directions in this topic.