TECHNISCHE
UNIVERSITÄT
DARMSTADT

Technical University Darmstadt

– Department of Computer Science –

Introduction to Cryptography

A Summary of the Course Contents

Author:

Jonas Weßner

Teaching professor	:	Prof. Dr. Marc Fischlin
Semester	:	Winter semester 2022/2023

CONTENTS

LIST OF ABBREVIATIONS

1

INTRODUCTION

Cryptography is the science of the processes for securing information, data and systems.

Typically we talk about certain key concepts, which differ from book to book. In this course, we define them as the following:

C Confidentiality: Attackers cannot read a message.

I Integrity: Attackers cannot modify the content of a message (in a narrow sense), cannot fake the message's sender address (authenticity) and receivers cannot claim, they have received the message (non-repudiation).

A Availability: A system is always functional (less relevant in this course)

Often in this lecture, we take a three-step approach to cryptographic processes:

1. **Abstract Object:** We define what interface our cryptographic process should work with. Often, we define one process for encryption, one for decryption and maybe others like one for key generation. We do not yet define their implementation.
2. **Security Model:** We define how our abstract object is used and in which cases our abstract object is secure. Questions to be answered are e.g. when is an attack successful and what is information the attacker is allowed to possess without introducing insecurity?
3. **Security Proof a Specific process:** When given a concrete implementation of our security model, we check the security of that model. For example, we prove that an attacker must calculate at least n hash sums to compromise the system.

Cryptographic processes can be classified in a matrix as follows:

	Private-Key	**Public-Key**
Confidentiality	Symmetric processes	Asymmetric processes
Integrity	Message Authentication Codes (MACs)	Digital signatures

Furthermore, there are elementary processes serving as building blocks to build cryptographic processes:

- Pseudo random number generators (PRNGs)
- Hash functions

- Blockcipher
- Number theory

2

PRIVATE-KEY CRYPTOGRAPHY

2.1 ONE TIME PAD

The One Time Pad works as follows:

1. Alice has a message $m \in \{0,1\}^n$ to be sent to Bob. Furthermore, Alice and Bob possess a common key $k \in \{0,1\}^n, n \in N$. n is called the security parameter.
2. Alice computes a cipher text $c \in \{0,1\}^n$ as $c = m \oplus k$ and sends it to Bob.
3. Bob reconstructs the plain message as $m = c \oplus k = m \oplus k \oplus k = m \oplus \{0\}^n$.

Therefore, the abstract object for this cryptographic process consists of the following functions[1]:

- $kGen(1^n) \rightarrow k \in \{0,1\}^n$
- $enc(m,k) \rightarrow c, \quad |m| = |k| = |c| = n$
- $dec(c,k) \rightarrow m \quad || \quad \perp$

The functional correctness is then defined as follows:
For all security parameter $n \in N$, for all messages m, for all keys $k \leftarrow kGen(1^n)$, for all keys $k \leftarrow kGen(m,k)$ and for all ciphertexts $c \leftarrow enc(m,k)$ applies $dec(c,k) = m$. That means that we must be able to decrypt all encrypted messages for all possible input parameters.

Security of Shannon's OTP

Independently of a bit m_i in the message m, the OTP flips this bit with the probability of $\frac{1}{2}$ creating a distribution of ciphtertexts c that is independent of the messages m. And because m and c are independent, it is intuitive that knowing c cannot reveal anything about m. We can also prove this mathematically because the probability for two independent events to take place is the product of the individual probabilities: $Pr[M = m|C = c] = \frac{Pr[M=m \wedge C=c]}{Pr[C=c]} = \frac{Pr[M=m] \cdot Pr[C=c]}{Pr[C=c]} = Pr[M = m]$.

1 The symbol $\perp$ is indicating an error.

In practice, perfect security is not feasible because this would mean that for transferring each message m a key k with at least the same length $|k| \geq |m|$ would have to be transferred (see definition of perfect security in section 3.2). For this reason, we define a weakened security definition, which comes in two variants – Concrete security and asymptotic security. In this course, we look at asymptotic security.

Concrete Security:	**Asymptotic Security**
A process is $(t, \mathcal{E})$-secure if no attacker A can break it with at most t steps with a probability of $\mathcal{E}$.	A process is secure, if no efficient (polynomial limited by security parameter n) algorithm breaks it with non-negligible (less than $\frac{1}{poly}$) probability.

The relevant terms are defined as follows, where n is the security parameter (e.g. key size):

- *Efficient Algorithm:* Algorithm with polynomial runtime with regard to n.
- *Non-negligible Probability:* An inversely polynomial probability ($\frac{1}{poly}$) with regard to n.
- *Negligible Probability:* Smaller than an inversely polynomial function (i.e. less than non-negligible) $\leq \frac{1}{poly}$. Mathematically speaking, a function $\mathcal{E} := N \to R$ if there can be found a value $limit \in N$, such that for all polynomial functions $poly$ applies $\mathcal{E}(n) \leq \frac{1}{poly} \quad | \quad n \geq limit$. This means that, if the security parameter is high enough, its value is smaller than any polynomial function. Stating that a function $\mathcal{E}$ is negligible can be written in three ways: $\mathcal{E} \leq neg(n)$, $\mathcal{E} = neg(n)$ or $\mathcal{E} \approx 0$. Furthermore, if a the function $\mathcal{E}_1$ and the function $\mathcal{E}_2$ differ by a negligible difference, we can write $\mathcal{E}_1 \approx \mathcal{E}_2$.

Examples for Negligibility of probability functions

- $\mathcal{E} = 2^{-n}$ is *negligible* because exponential functions grow faster than polynomial functions.
- $\mathcal{E} = n^{-5}$ is *not negligible* because there are polynomials that have smaller values as n approaches infinity e.g. n^{-6}.
- $\mathcal{E} = \begin{cases} 2^{-n} & \text{if } n \text{ is even} \\ n^{-5} & \text{if } n \text{ is uneven} \end{cases}$ is *not negligible* because for some values (all uneven values) it behaves like a polynomial function, such that e.g. n^{-6} would have greater values regardless of a chosen $limit$.

- $\mathcal{E} = \frac{1}{8}$ is *not negligible* because it does not approach zero and therefore there are many polynomial functions that are smaller than $\mathcal{E}$ for high n

Arithmetic Laws for Negligible Functions

If $mathcal E_1$ and $mathcal E_2$ are negligible functions $mathcal E_1, mathcal E_2 \approx 0$ and q is a polynomial function $q = poly$ and ω ia a non-negligible $\omega \not\approx 0$ function, then applies:

$$\mathcal{E}_1(n) + \mathcal{E}_2(n) \approx 0 \tag{2.1}$$
$$\mathcal{E}_1(n) \cdot q(n) \approx 0 \tag{2.2}$$
$$\omega(n) - \mathcal{E}_1(n) \not\approx 0 \tag{2.3}$$

More intuitively phrased:

(2.1) Executing a negligible function and another negligible function after each other will still result in something negligible.

(2.1) Executing a negligible function for any polynomial number of times will still result in a negligible value in sum because the negligible function approaches zero faster than any polynomial function could grow towards positive values.

(2.1) Subtracting something negligible from something non-negligible does not make a difference – the result is still non-negligible.

2.2.1 *Semantic Security – A Simulation-Based Definition*

A cryptographic process is semantically secure if for all PPT attacker algorithms A, which calculate information $f(m)$ about a length-invariant message m (message of given length n) with the help of the message's ciphertext c with a certain probability, there is a simulator algorithm S, which can compute the same information with a negligibly different probability:

$$Pr[A(1^n, c) = f(m)] \approx Pr[S(1^n) = f(m)]$$

More intuitively speaking, everything which can be computed with the knowledge about the ciphtertext of a message can also be computed with nearly the same precision without knowledge about the ciphertext.

This is the simulation-based definition of asymptotic security.

2.2.2 *Indistinguishability (IND) Security – a Game-Based Definition*

This is a game-based definition of asymptotic security, which is equivalent to the semantic security definition (2.2.1).

A cryptographic process $\mathcal{E}$ is indistinguishable (IND) if for all PPT attackers A applies:

$$Pr[Exp^{IND}_{\mathcal{E},A}(n) = 1] \approx \frac{1}{2}$$

, where $Exp^{IND}_{\mathcal{E},A}(n)$ is defined as the following algorithm:

$$KGen(1^n) \to k \tag{2.4}$$
$$\{0,1\} \to b \tag{2.5}$$
$$A(1^n) \to (st, m_0, m_1)| \quad |m_0| = |m_1| = n \tag{2.6}$$
$$Enc(m_b, k) \to c \tag{2.7}$$
$$A(1^n, st, c) \to a \tag{2.8}$$
$$return \quad a == b \tag{2.9}$$

More intuitive worded, this means that (2.4) a secret key of length n is chosen (2.5) a secret random bit is chosen (2.6) the attacker selects two messages of his choice (2.7) one through b randomly selected message is encrypted (2.8) the attacker looks at the ciphertext of one of its messages and decides which messages ciphertext that is and (2.9) the experiment returns true if the attacker has been able to identify the message and otherwise false. If the output of this experiment is negligible close to $\frac{1}{2}$ for all attackers A, then the attackers have no significant advantage over simply guessing and, therefore, the cryptographic process $\mathcal{E}$ is secure.

An example of an insecure process $\mathcal{E}$ could be an algorithm, which simply flips all bits: $Enc(m,k) =\sim (m)$. The attacker could then just use the Enc algorithm (which is not secret) with any key and get the same result because the Enc algorithm does not use the key. Therefore, he could distinguish messages with the probability of $1 \not\approx \frac{1}{2}$.

An example of a secure algorithm is Shannon's OTP. The reason is that without knowing the key, the attacker cannot distinguish two messages, because the OTP flips bits with the probability of $\frac{1}{2}$, such that the cipthertext is completely random, just like the key.

2.3 Pseudo Random (Number) Generators (PR(N)G)

An algorithm G is a secure PRG, if for all PPT attackers A applies:

$$Pr[Exp^{PRG}_{\mathcal{G},\mathcal{D},l,A}(n) = 1] \approx \frac{1}{2}$$

, where $Exp^{PRG}_{\mathcal{G},\mathcal{D},l,A}(n)$ is defined as:

$$KGen(1^n) \rightarrow k \tag{2.10}$$
$$\{0,1\} \rightarrow b \tag{2.11}$$
$$G(k) \rightarrow y_0 \tag{2.12}$$
$$\{0,1\}^{|y_0|} \rightarrow \tag{2.13}$$
$$A(1^n, y_b) \rightarrow a \tag{2.14}$$
$$return \quad a == b \tag{2.15}$$

More intuitively speaking, this means if we generate a pseudo-random bitstring y_0 using G and a fully random bitstring y_1, no attacker algorithm can distinguish them.

3

GLOSSARY

In this chapter, basic terms will be defined and explained.

3.1 KERCKHOFFS'S PRINCIPLE

Security should not require the system, but merely the key to be secret.

3.2 PERFECT SECURITY

A cryptographic process is perfectly secure if for all messages in the set of possibles messages $m \in M$ and all possibly producible ciphertexts $c \in C \quad | \quad Pr[C = c] > 0$ the probability that m occurs is equal no matter if the ciphertext is known or unknown: $Pr[M = m] = Pr[M = m|C = c]$. More simply put, this means that an attacker cannot gain any knowledge about the message when seeing its ciphertext regardless of the message's and the ciphertext's concrete values.

3.3 CONDITIONAL PROBABILITY

Conditional probability $Pr[A = a|B = b]$ is the probability that the event $a \in A$ occurs if it is already known that the event $b \in B$ occurs. It is calculated as $Pr[A = a|B = b] = \frac{Pr[A=a] \wedge Pr[B=b]]}{Pr[B=b]}$. An example is a probability rolling a 6-sided dice results in the number 6 if it is already known that the result is an even number: $Pr[is6|iseven] = \frac{Pr[is6] \wedge Pr[iseven]}{iseven} = \frac{\frac{1}{6}}{\frac{1}{2}} = \frac{1}{3}$.

Perfect security can only be reached for processes with deterministic a decryption function if the key space is at least as big as the message space $|K| \geq |M|$. This is because then a given ciphtertext c could be decrypted to $|K|$ different messages m (determinism), which minimizes the search space for an attacker from $|M|$ possibilities to $|K|$ possibilities and therefore also decrease the attacker's uncertainty..

3.4 HIDING MESSAGE LENGTH

Completely hiding the length of a message is impossible if the length of possible messages is unlimited. Hiding messages' lengths would require changing their length for transmission. The message length cannot be decreased because this would result in data loss. Therefore, messages would have to be extended up to at least the size of the longest possible message to make all messages appear to have the same length. If the message size is unlimited, we cannot expand messages to that nonexisting limit.

3.5 INSECURITY OF SHIFT-CIPHERS

Shift-ciphers, like Ceasar, are not perfectly secure, which can be shown by contraposition: If the possible messages are defined as $M = \{aa, ab\}$, a shift cipher shifting each character by some number of characters in an alphabet would always create ciphertexts as follows $Dec(aa) \rightarrow c_0c0$ and $Dec(ab) \rightarrow c_0c_1 \quad | \quad c_0 \neq c_1$ Therefore, an attacker's probability when seeing the ciphertexts to guess the correct message $m \in M$ is equal to 1 and greater than the initial probability, which was $\frac{1}{2}$. A concrete example would be: $Pr[M = aa|C = c_0c_0] = 1 \neq Pr[M = aa] = \frac{1}{2}$. We see that limiting the set of possible messages can be a helpful method for contradictions of perfect correctness.