Christof Paar
Jan Pelzl

Understanding Cryptography

A Textbook for Students and Practitioners

Springer

Christof Paar · Jan Pelzl

Understanding Cryptography

A Textbook for Students and Practitioners

Foreword by Bart Preneel

Springer

Prof. Dr.-Ing. Christof Paar
Chair for Embedded Security
Department of Electrical Engineering
and Information Sciences
Ruhr-Universität Bochum
44780 Bochum
Germany
cpaar@crypto.rub.de

Dr.-Ing. Jan Pelzl
escrypt GmbH – Embedded Security
Zentrum für IT-Sicherheit
Lise-Meitner-Allee 4
44801 Bochum
Germany
jpelzl@escrypt.com

ISBN 978-3-642-04100-6 e-ISBN 978-3-642-04101-3
DOI 10.1007/978-3-642-04101-3
Springer Heidelberg Dordrecht London New York

ACM Computing Classification (1998): E.3, K.4.4, K.6.5.

Library of Congress Control Number: 2009940447

Cover design: KuenkelLopka GmbH

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

To

Flora, Maja, Noah and Sarah

as well as to

Karl, Greta and Nele

While writing this book we noticed that for some reason the names of our spouses and children are limited to five letters. As far as we know, this has no cryptographic relevance.

Foreword

Academic research in cryptology started in the mid-1970s; today it is a mature research discipline with an established professional organization (IACR, International Association for Cryptologic Research), thousands of researchers, and dozens of international conferences. Every year more than a thousand scientific papers are published on cryptology and its applications.

Until the 1970s, cryptography was almost exclusively found in diplomatic, military and government applications. During the 1980s, the financial and telecommunications industries deployed hardware cryptographic devices. The first mass-market cryptographic application was the digital mobile phone system of the late 1980s. Today, everyone uses cryptography on a daily basis: Examples include unlocking a car or garage door with a remote-control device, connecting to a wireless LAN, buying goods with a credit or debit card in a brick and mortar store or on the Internet, installing a software update, making a phone call via voice-over-IP, or paying for a ride on a public transport system. There is no doubt that emerging application areas such as e-health, car telematics and smart buildings will make cryptography even more ubiquitous.

Cryptology is a fascinating discipline at the intersection of computer science, mathematics and electrical engineering. As cryptology is moving fast, it is hard to keep up with all the developments. During the last 25 years, the theoretical foundations of the area have been strengthened; we now have a solid understanding of security definitions and of ways to prove constructions secure. Also in the area of applied cryptography we witness very fast developments: old algorithms are broken and withdrawn and new algorithms and protocols emerge.

While several excellent textbooks on cryptology have been published in the last decade, they tend to focus on readers with a strong mathematical background. Moreover, the exciting new developments and advanced protocols form a temptation to add ever more fancy material. It is the great merit of this textbook that it restricts itself to those topics that are relevant to practitioners today. Moreover, the mathematical background and formalism is limited to what is strictly necessary and it is introduced exactly in the place where it is needed. This “less is more” approach is very suitable to address the needs of newcomers in the field, as they get introduced

step by step to the basic concepts and judiciously chosen algorithms and protocols. Each chapter contains very helpful pointers to further reading, for those who want to expand and deepen their knowledge.

Overall, I am very pleased that the authors have succeeded in creating a highly valuable introduction to the subject of applied cryptography. I hope that it can serve as a guide for practitioners to build more secure systems based on cryptography, and as a stepping stone for future researchers to explore the exciting world of cryptography and its applications.

Leuven, August 2009 *Bart Preneel*

Preface

Cryptography has crept into everything, from Web browsers and e-mail programs to cell phones, bank cards, cars and even into medical implants. In the near future we will see many new exciting applications for cryptography such as radio frequency identification (RFID) tags for anti-counterfeiting or car-to-car communications (we've worked on securing both of these applications). This is quite a change from the past, where cryptography had been traditionally confined to very specific applications, especially government communications and banking systems. As a consequence of the pervasiveness of crypto algorithms, an increasing number of people must understand how they work and how they can be applied in practice. This book addresses this issue by providing a comprehensive introduction to modern applied cryptography that is equally suited for students and practitioners in industry.

Our book provides the reader with a deep understanding of how modern cryptographic schemes work. We introduce the necessary mathematical concepts in a way that is accessible for every reader with a minimum background in college-level calculus. It is thus equally well suited as a textbook for undergraduate or beginning graduate classes, or as a reference book for practicing engineers and computer scientists who are interested in a solid understanding of modern cryptography.

The book has many features that make it a unique source for practitioners and students. We focused on practical relevance by introducing most crypto algorithms that are used in modern real-world applications. For every crypto scheme, up-to-date security estimations and key length recommendations are given. We also discuss the important issue of software and hardware implementation for every algorithm. In addition to crypto algorithms, we introduce topics such as important cryptographic protocols, modes of operation, security services and key establishment techniques. Many very timely topics, e.g., lightweight ciphers which are optimized for constrained applications (such as RFID tags or smart cards) or new modes of operations, are also contained in the book.

A discussion section at the end of each chapter with annotated references provides plenty of material for further reading. For classroom use, these sections are

an excellent source for course projects. In particular, when used as a textbook, the companion website for the book is highly recommended:

`www.crypto-textbook.com`

Readers will find many ideas for course projects, links to open-source software, test vectors, and much more information on contemporary cryptography. In addition, links to video lectures are provided.

How to Use the Book

The material in this book has evolved over many years and is "classroom proven". We've taught it both as a course for beginning graduate students and advanced undergraduate students and as a pure undergraduate course for students majoring in our IT security programs. We found that one can teach most of the book content in a two-semester course, with 90 minutes of lecture time plus 45 minutes of help session with exercises per week (total of 10 ECTS credits). In a typical US-style three-credit course, or in a one-semester European course, some of the material should be omitted. Here are some reasonable choices for a one-semester course:

Curriculum 1 Focus on the *application of cryptography*, e.g., in a computer science or electrical engineering program. This crypto course is a good addition to courses in computer networks or more advanced security courses: Chap. 1; Sects. 2.1–2.2; Chap. 4; Sect. 5.1; Chap. 6; Sects. 7.1–7.3; Sects. 8.1–8.4; Sects. 10.1–10.2; Chap. 11; Chap. 12; and Chap. 13.

Curriculum 2 Focus on *cryptographic algorithms and their mathematical background*, e.g., as an applied cryptography course in computer science, electrical engineering or in an (undergraduate) math program. This crypto course works also nicely as preparation for a more theoretical graduate courses in cryptography: Chap. 1; Chap. 2; Chap. 3; Chap. 4; Chap. 6; Chap. 7; Sects. 8.1 – 8.4; Chap. 9; Chap. 10; and Sects. 11.1 – 11.2.

Trained as engineers, we have worked in applied cryptography and security for more than 15 years and hope that the readers will have as much fun with this fascinating field as we've had!

Bochum, *Christof Paar*
September 2009 *Jan Pelzl*

Acknowledgements

Writing this book would have been impossible without the help of many people. We hope we did not forget anyone in our list.

We are grateful for the excellent work of Daehyun Strobel and Pascal Wißmann, who provided most of the artwork in the book and never complained about our many changes. Axel Poschmann provided the section about the PRESENT block cipher, a very timely topic, and we are thankful for his excellent work. Help with technical questions was provided by Frederick Armknecht (stream ciphers), Roberto Avanzi (finite fields and elliptic curves), Alexander May (number theory), Alfred Menezes and Neal Koblitz (history of elliptic curve cryptography), Matt Robshaw (AES), and Damian Weber (discrete logarithms).

Many thanks go the members of the Embedded Security group at the University of Bochum — Andrey Bogdanov, Benedikt Driessen, Thomas Eisenbarth, Tim Güneysu, Stefan Heyse, Markus Kasper, Timo Kasper, Amir Moradi and Daehyun Strobel — who did much of the technical proofreading and provided numerous suggestions for improving the presentation of the material. Special thanks to Daehyun for helping with examples and some advanced LaTeX work, and to Markus for his help with problems. Olga Paustjan's help with artwork and typesetting is also very much appreciated.

An earlier generation of doctoral students from our group — Sandeep Kumar, Kerstin Lemke-Rust, Andy Rupp, Kai Schramm, and Marko Wolf — helped to create an online course that covered similar material. Their work was very useful and was a great inspiration when writing the book.

Bart Preneel's willingness to provide the Foreword is a great honor for us and we would like to thank him at this point again. Last but not least, we thank the people from Springer for their support and encouragement. In particular, thanks to our editor Ronan Nugent and to Alfred Hofmann.

Table of Contents

Chapter 1
Introduction to Cryptography and Data Security

This section will introduce the most important terms of modern cryptology and will teach an important lesson about proprietary vs. openly known algorithms. We will also introduce modular arithmetic which is also of major importance in public-key cryptography.

In this chapter you will learn:

- The general rules of cryptography
- Key lengths for short-, medium- and long-term security
- The difference between different types of attacks against ciphers
- A few historical ciphers, and on the way we will learn about modular arithmetic, which is of major importance for modern cryptography as well
- Why one should only use well-established encryption algorithms

C. Paar, J. Pelzl, *Understanding Cryptography*,
DOI 10.1007/978-3-642-04101-3_1,

1.1 Overview of Cryptology (and This Book)

If we hear the word *cryptography* our first associations might be e-mail encryption, secure website access, smart cards for banking applications or code breaking during World War II, such as the famous attack against the German Enigma encryption machine (Fig. 1.1).

Fig. 1.1 The German Enigma encryption machine (reproduced with permission from the Deutsches Museum, Munich)

Cryptography seems closely linked to modern electronic communication. However, cryptography is a rather old business, with early examples dating back to about 2000 B.C., when non-standard "secret" hieroglyphics were used in ancient Egypt. Since Egyptian days cryptography has been used in one form or the other in many, if not most, cultures that developed written language. For instance, there are documented cases of secret writing in ancient Greece, namely the *scytale* of Sparta (Fig. 1.2), or the famous Caesar cipher in ancient Rome, about which we will learn later in this chapter. This book, however, strongly focuses on modern cryptographic

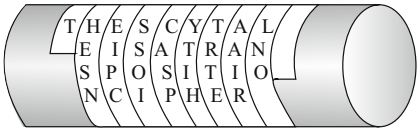

Fig. 1.2 Scytale of Sparta

methods and also teaches many data security issues and their relationship with cryptography.

Let's now have a look at the field of *cryptography* (Fig. 1.3). The first thing

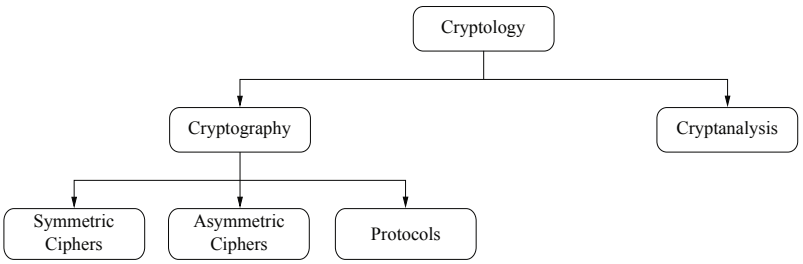

Fig. 1.3 Overview of the field of cryptology

that we notice is that the most general term is *cryptology* and not *cryptography*. Cryptology splits into two main branches:

Cryptography is the science of secret writing with the goal of hiding the meaning of a message.

Cryptanalysis is the science and sometimes art of *breaking* cryptosystems. You might think that code breaking is for the intelligence community or perhaps organized crime, and should not be included in a serious classification of a scientific discipline. However, most cryptanalysis is done by respectable researchers in academia nowadays. Cryptanalysis is of central importance for modern cryptosystems: without people who try to break our crypto methods, we will never know whether they are really secure or not. See Sect. 1.3 for more discussion about this issue.

Because cryptanalysis is the only way to assure that a cryptosystem is secure, it is an integral part of cryptology. Nevertheless, the focus of this book is on crypto**graphy**: We introduce most important practical crypto algorithms in detail. These are all crypto algorithms that have withstood cryptanalysis for a long time, in most cases for several decades. In the case of crypt**analysis** we will mainly restrict ourselves to providing state-of-the-art results with respect to breaking the crypto algorithms that are introduced, e.g., the factoring record for breaking the RSA scheme.

Let's now go back to Fig. 1.3. Cryptography itself splits into three main branches:

Symmetric Algorithms are what many people assume cryptography is about: two parties have an encryption and decryption method for which they share a secret key. All cryptography from ancient times until 1976 was exclusively based on symmetric methods. Symmetric ciphers are still in widespread use, especially for data encryption and integrity check of messages.

Asymmetric (or Public-Key) Algorithms In 1976 an entirely different type of cipher was introduced by Whitfield Diffie, Martin Hellman and Ralph Merkle. In public-key cryptography, a user possesses a secret key as in symmetric cryptography but also a public key. Asymmetric algorithms can be used for applications such as digital signatures and key establishment, and also for classical data encryption.

Cryptographic Protocols Roughly speaking, crypto protocols deal with the application of cryptographic algorithms. Symmetric and asymmetric algorithms

can be viewed as building blocks with which applications such as secure Internet communication can be realized. The Transport Layer Security (TLS) scheme, which is used in every Web browser, is an example of a cryptographic protocol.

Strictly speaking, hash functions, which will be introduced in Chap. 11, form a third class of algorithms but at the same time they share some properties with symmetric ciphers.

In the majority of cryptographic applications in practical systems, symmetric and asymmetric algorithms (and often also hash functions) are all used together. This is sometimes referred to as *hybrid schemes*. The reason for using both families of algorithms is that each has specific strengths and weaknesses.

The main focus of this book is on symmetric and asymmetric algorithms, as well as hash functions. However, we will also introduce basic security protocols. In particular, we will introduce several key establishment protocols and what can be achieved with crypto protocols: confidentiality of data, integrity of data, authentication of data, user identification, etc.

1.2 Symmetric Cryptography

This section deals with the concepts of symmetric ciphers and it introduces the historic substitution cipher. Using the substitution cipher as an example, we will learn the difference between brute-force and analytical attacks.

1.2.1 Basics

Symmetric cryptographic schemes are also referred to as *symmetric-key*, *secret-key*, and *single-key* schemes or algorithms. Symmetric cryptography is best introduced with an easy to understand problem: There are two users, Alice and Bob, who want to communicate over an insecure *channel* (Fig. 1.4). The term channel might sound a bit abstract but it is just a general term for the communication link: This can be the Internet, a stretch of air in the case of mobile phones or wireless LAN communication, or any other communication media you can think of. The actual problem starts with the bad guy, Oscar[1], who has access to the channel, for instance, by hacking into an Internet router or by listening to the radio signals of a Wi-Fi communication. This type of unauthorized listening is called *eavesdropping*. Obviously, there are many situations in which Alice and Bob would prefer to communicate without Oscar listening. For instance, if Alice and Bob represent two offices of a car manufacturer, and they are transmitting documents containing the business strategy for the introduction of new car models in the next few years, these documents should

[1] The name Oscar was chosen to remind us of the word opponent.

not get into the hands of their competitors, or of foreign intelligence agencies for that matter.

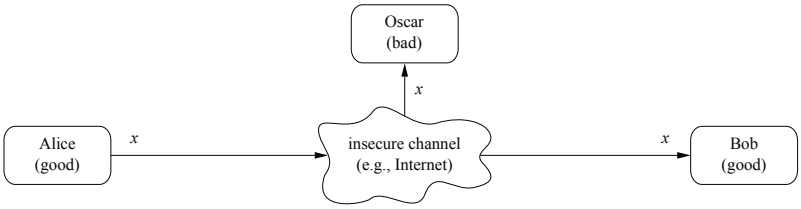

Fig. 1.4 Communication over an insecure channel

In this situation, symmetric cryptography offers a powerful solution: Alice encrypts her message x using a symmetric algorithm, yielding the ciphertext y. Bob receives the ciphertext and decrypts the message. Decryption is, thus, the inverse process of encryption (Fig. 1.5). What is the advantage? If we have a strong encryption algorithm, the ciphertext will look like random bits to Oscar and will contain no information whatsoever that is useful to him.

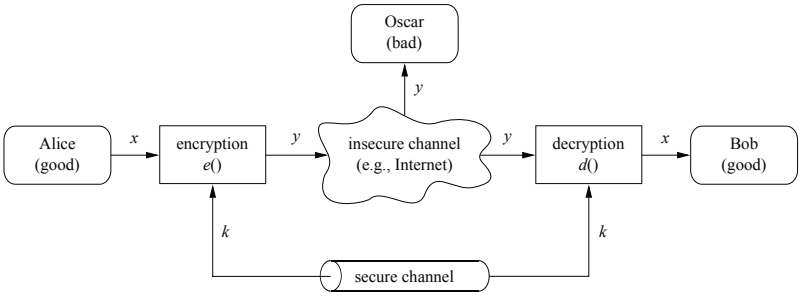

Fig. 1.5 Symmetric-key cryptosystem

The variables x, y and k in Fig. 1.5 are important in cryptography and have special names:

- x is called *plaintext* or *cleartext*,
- y is called *ciphertext*,
- k is called the *key*,
- the set of all possible keys is called the *key space*.

The system needs a secure channel for distribution of the key between Alice and Bob. The secure channel shown in Fig. 1.5 can, for instance, be a human who is transporting the key in a wallet between Alice and Bob. This is, of course, a somewhat cumbersome method. An example where this method works nicely is the pre-shared keys used in Wi-Fi Protected Access (WPA) encryption in wireless

LANs. Later in this book we will learn methods for establishing keys over insecure channels. In any case, the key has only to be transmitted once between Alice and Bob and can then be used for securing many subsequent communications.

One important and also counterintuitive fact in this situation is that both the encryption and the decryption algorithms are publicly known. It seems that keeping the *encryption algorithm* secret should make the whole system harder to break. However, secret algorithms also mean untested algorithms: The only way to find out whether an encryption method is strong, i.e., cannot be broken by a determined attacker, is to make it public and have it analyzed by other cryptographers. Please see Sect. 1.3 for more discussion on this topic. The only thing that should be kept secret in a sound cryptosystem is the key.

Remarks:

1. Of course, if Oscar gets hold of the key, he can easily decrypt the message since the algorithm is publicly known. Hence it is crucial to note that the problem of transmitting a message securely is reduced to the problems of transmitting a key secretly and of storing the key in a secure fashion.
2. In this scenario we only consider the problem of confidentiality, that is, of hiding the contents of the message from an eavesdropper. We will see later in this book that there are many other things we can do with cryptography, such as preventing Oscar from making unnoticed changes to the message (message integrity) or assuring that a message really comes from Alice (sender authentication).

1.2.2 Simple Symmetric Encryption: The Substitution Cipher

We will now learn one of the simplest methods for encrypting text, the *substitution (= replacement) cipher*. Historically this type of cipher has been used many times, and it is a good illustration of basic cryptography. We will use the substitution cipher for learning some important facts about key lengths and about different ways of attacking ciphers.

The goal of the substitution cipher is the encryption of text (as opposed to bits in modern digital systems). The idea is very simple: We substitute each letter of the alphabet with another one.

Example 1.1.

$$\begin{aligned} \texttt{A} &\rightarrow \texttt{k} \\ \texttt{B} &\rightarrow \texttt{d} \\ \texttt{C} &\rightarrow \texttt{w} \\ &\ldots \end{aligned}$$

For instance, the pop group `ABBA` would be encrypted as `kddk`.

◇

We assume that we choose the substitution table completely randomly, so that an attacker is not able to guess it. Note that the substitution table is the key of this cryptosystem. As always in symmetric cryptography, the key has to be distributed between Alice and Bob in a secure fashion.

Example 1.2. Let's look at another ciphertext:

```
iq ifcc vqqr fb rdq vfllcq na rdq cfjwhwz hr bnnb
            hcc hwwhbsqvqbre hwq vhlq
```

◇

This does not seem to make too much sense and looks like decent cryptography. *However, the substitution cipher is not secure at all!* Let's look at ways of breaking the cipher.

First Attack: Brute-Force or Exhaustive Key Search

Brute-force attacks are based on a simple concept: Oscar, the attacker, has the ciphertext from eavesdropping on the channel and happens to have a short piece of plaintext, e.g., the header of a file that was encrypted. Oscar now simply decrypts the first piece of ciphertext with *all possible* keys. Again, the key for this cipher is the substitution table. If the resulting plaintext matches the short piece of plaintext, he knows that he has found the correct key.

Definition 1.2.1 Basic Exhaustive Key Search or Brute-force Attack
Let (x,y) denote the pair of plaintext and ciphertext, and let $K = \{k_1,\ldots,k_\kappa\}$ be the key space of all possible keys k_i. A brute-force attack now checks for every $k_i \in K$ if

$$d_{k_i}(y) \stackrel{?}{=} x.$$

If the equality holds, a possible correct key is found; if not, proceed with the next key.

In practice, a brute-force attack can be more complicated because incorrect keys can give false positive results. We will address this issue in Sect. 5.2.

It is important to note that a brute-force attack against symmetric ciphers is always possible *in principle*. Whether it is feasible in practice depends on the key space, i.e., on the number of possible keys that exist for a given cipher. If testing all the keys on many modern computers takes too much time, i.e., several decades, the cipher is *computationally secure* against a brute-force attack.

Let's determine the key space of the substitution cipher: When choosing the replacement for the first letter A, we randomly choose one letter from the 26 letters of the alphabet (in the example above we chose k). The replacement for the next alphabet letter B was randomly chosen from the remaining 25 letters, etc. Thus there exist the following number of different substitution tables:

$$\text{key space of the substitution cipher} = 26 \cdot 25 \cdots 3 \cdot 2 \cdot 1 = 26! \approx 2^{88}$$

Even with hundreds of thousands of high-end PCs such a search would take several decades! Thus, we are tempted to conclude that the substitution cipher is secure. But this is incorrect because there is another, more powerful attack.

Second Attack: Letter Frequency Analysis

First we note that the brute-force attack from above treats the cipher as a black box, i.e., we do not analyze the internal structure of the cipher. The substitution cipher can easily be broken by such an analytical attack.

The major weakness of the cipher is that each plaintext symbol always maps to the same ciphertext symbol. That means that the statistical properties of the plaintext are preserved in the ciphertext. If we go back to the second example we observe that the letter q occurs most frequently in the text. From this we know that q must be the substitution for one of the frequent letters in the English language.

For practical attacks, the following properties of language can be exploited:

1. Determine the frequency of every ciphertext letter. The frequency distribution, often even of relatively short pieces of encrypted text, will be close to that of the given language in general. In particular, the most frequent letters can often easily be spotted in ciphertexts. For instance, in English E is the most frequent letter (about 13%), T is the second most frequent letter (about 9%), A is the third most frequent letter (about 8%), and so on. Table 1.1 lists the letter frequency distribution of English.
2. The method above can be generalized by looking at pairs or triples, or quadruples, and so on of ciphertext symbols. For instance, in English (and some other European languages), the letter Q is almost always followed by a U. This behavior can be exploited to detect the substitution of the letter Q and the letter U.
3. If we assume that word separators (blanks) have been found (which is only sometimes the case), one can often detect frequent short words such as THE, AND, etc. Once we have identified one of these words, we immediately know three letters (or whatever the length of the word is) for the entire text.

In practice, the three techniques listed above are often combined to break substitution ciphers.

Example 1.3. If we analyze the encrypted text from Example 1.2, we obtain:

```
WE WILL MEET IN THE MIDDLE OF THE LIBRARY AT NOON
            ALL ARRANGEMENTS ARE MADE
```