Preface

The primary goal of this book is to present the basic concepts in computer and network security. The starting point for the book was the lecture notes that were used for teaching the undergraduate course on computer security at the University of Wollongong (Australia). Later more topics were added to the book. These topics were mainly taught to postgraduate students as an advanced cryptography course. Some chapters, especially those towards the end of the book, were included to help students in their seminar presentations. The book was recently used as the textbook for a new course, Cryptography and Information Security, offered for third-year students in the Computing Department at Macquarie University (Australia).

The book contains 18 chapters. It starts with an introductory chapter (Chap. 1) that gives a brief summary of the history of cryptography. As the book is meant to be self-contained, the necessary background knowledge is given in the theory chapter (Chap. 2). It starts with elements of number theory, goes through algebraic structures, the complexity of computing, and elements of information theory.

The chapter on private-key cryptosystems (Chap. 3) covers classical ciphers, the DES family of ciphers, and a selected subset of modern cryptosystems submitted for the AES call. An introduction to differential and linear cryptanalysis closes the chapter. The next chapter (Chap. 4) sets the background for public-key cryptography. It describes the concept of a public key and discusses its implementations. The RSA cryptosystem deserves special attention for at least two reasons. The first one is that its security nicely relates to the difficulty of factoring. The other one is the widespread use of RSA for communication security. Probabilistic encryption and modern public-key cryptosystems close the chapter.

In Chap. 5, pseudorandomness is studied and the concept of polynomial indistinguishability is introduced. Next, pseudorandom generators of bits, functions, and permutations are described. Hashing is studied in Chap. 6. The birthday paradox is presented and its application for breaking hash functions explored. The main part of the chapter is devoted to the MD family of hash functions (MD5, SHA-1, RIPEMD-160, and HAVAL). Keyed hashing closes the chapter.

Chapter 7 deals with signature schemes and starts with one-time signature schemes. The basic signature schemes examined in this chapter are the RSA and ElGamal signatures. In addition, the chapter discusses blind, undeniable, and fail-stop signatures. Authentication, although related to digital signatures, has developed its own theory and vocabulary, which are presented in Chap. 8.

Secret sharing is one of the main cryptographic tools that enables groups to perform cryptographic operations. The basic theory of secret sharing is given in Chap. 9 and the application of secret sharing in cryptography is presented in Chap. 10.

The key establishment protocols discussed in Chap. 11 can be split into two broad categories: key-agreement and key-distribution protocols. These two categories normally relate to the case when two parties wish to generate a fresh and secret key. Multiparty versions of key establishment protocols are of growing interest and are also presented in the chapter. A short exposition of the BAN logic concludes the chapter.

Zero-knowledge proof systems are interactive systems in which two parties – the prover and verifier – interact. The prover claims that a statement is true and the verifier wants to be convinced that it is true. The parties interact, and at the end of the interaction the verifier is either convinced that the statement is true or, alternatively, the verifier discovers that the statement is not true. This topic is studied in Chap. 12.

Identification is discussed in Chap. 13. The discussion starts with a review of biometric techniques applied to user identification. Next, passwords and challenge-response identification are examined. The main part of the chapter is, however, devoted to identification based on zero-knowledge proofs.

The chapter on intrusion detection (Chap. 14) first discusses two generic approaches: anomaly and misuse detection. The former approach makes decisions about a suspected intrusion using the profile of the behavior of the user. The latter approach uses a single profile that characterizes misuse of the computing resources. Selected implementations of host and network intrusion detection systems are also examined.

Chapter 15 is an introduction to the technical aspects of e-commerce and it covers electronic elections and electronic money. Chapter 16 looks at database security with an emphasis on security filters, cryptographic methods, and database views. A review of security features applied in Oracle8 concludes the chapter.

Chapter 17 explores access control. Three models are considered: mandatory, discretionary, and role-based access control. Some selected implementations are described as well. Chapter 18 deals with the IPSec protocol and computer viruses.

The contents of the book can be roughly divided into two parts:

1. Cryptography (Chaps. 3 to 13)
2. Computer and Network Security (Chaps. 14 to 18)

The chapters can be arranged into the following hierarchy:

Introduction	Background Theory

Cryptography			
Private-key Cryptosystems	Public-key Cryptosystems	Secret Sharing	Authentication
Pseudorandomness	Hashing	Zero-knowledge Proof Systems	
Digital Signatures			
Group-Oriented Cryptography	Key Establishment Protocols	Identification	

Computer and Network Security				
Intrusion Detection	Electronic Elections and Digital Money	Database Protection and Security	Access Control	Network Security

The cryptography part includes four basic chapters: private-key cryptosystems, public-key cryptosystems, secret sharing, and authentication. They can be studied independently as they overlap very little (see the third row of the above table). Chapters on pseudorandomness, hashing, and zero-knowledge proof systems (the fourth row of the table) build on knowledge of the material given in the basic chapters. For instance, zero-knowledge proof systems require a good grasp of public-key cryptosystems. The hashing chapter is less dependent on basic chapters but, certainly, hashing uses both private- and public-key cryptography. Digital signature concepts are tightly coupled with private- and public-key cryptography, and with hashing. The bottom chapters of the cryptography part include chapters on complex topics and the reader is encouraged to be sure that the concepts in the chapters in the two preceding rows are first understood. In particular, to follow material in the chapter on

- group-oriented cryptography, working through the chapters on public-key cryptosystems, secret sharing, hashing, and digital signatures is recommended;
- key establishment protocols, the reader is required to know public-key cryptography, including the theory behind digital signatures;
- identification, it is advisable to know digital signatures and zero-knowledge proof systems.

The book can be used as a text for undergraduate and postgraduate courses. The list below gives examples of possible courses that can be supported by the book:

- Introduction to Cryptography: Background Theory, Private-Key Cryptosystems, Public-Key Cryptosystems, Hashing, and Digital Signatures.
- Electronic Commerce: Background Theory, Public-Key Cryptosystems, Hashing, Digital Signatures, Electronic Elections, and Digital Money.
- Advanced Cryptography: Authentication, Secret Sharing, Group-Oriented Cryptography, Key Establishment Protocols, Zero-Knowledge Proof Systems, and Identification.
- Computer and Network Security: Intrusion Detection, Database Protection and Security, Access Control, and Network Security.

The contributions to the book are shown in the following table:

Coauthor Name	Chapters
Josef Pieprzyk	All chapters
Thomas Hardjono	14 and 16
Jennifer Seberry	3 and 18

Acknowledgements The authors are grateful to the persons who provided comments and constructive criticism. We are especially thankful to

Marc Fischlin, Ron Rivest, and Abhi Shelat

for their critical reviews, suggestions, and improvements. We also indebted to many colleagues who devoted their time and effort to give us their feedback. The list includes:

Colin Boyd
Nicolas Courtois
Ivo Desmedt
Dieter Gollmann
Hossein Ghodosi
Jeffrey Horton
Andrew Klapper
Keith Martin
Anish Mathuria
Krystian Matusiewicz
Igor Shparlinski
Michał Sramka
Janusz Stokłosa
Huaxiong Wang
Xianmo Zhang

We apologize to those colleagues whose names have inadvertently been dropped from the list.

The project would never have happened without the strong support and encouragement from Alfred Hofmann and his team at Springer-Verlag, including Ingeborg Mayer, Frank Holzwarth and Ronan Nugent. Thank you. We also thank Kate Krastev and Deanne van der Myle for their help in getting the book into its final form.

Readers who spot any omissions or errors are requested to contact the authors with details.

November 2002 Josef Pieprzyk, Thomas Hardjono, and Jennifer Seberry

Table of Contents

1 Introduction

1.1 Preamble

In 1988, the *New Encyclopædia Britannica* defined *cryptology* as:

> The science concerned with communications in secure and usually secret form. It encompasses both cryptography and cryptanalysis. The former involves the study and application of the principles and techniques by which information is rendered unintelligible to all but the intended receiver, while the latter is the science and art of solving cryptosystems to recover such information.

Today this definition needs to be extended as modern cryptology focuses its attention on the design and evaluation of a wide range of methods and techniques for information protection. Information protection covers not only secrecy (a traditional protection against eavesdropping) but also authentication, integrity, verifiability, nonrepudiation and other more specific security goals. The part of cryptology that deals with the design of algorithms, protocols, and systems which are used to protect information against specific threats is called *cryptography*.

To incorporate information protection into a system, protocol, or service, the designer needs to know:

- a detailed specification of the environment in which the system (protocol or service) is going to work, including a collection of security goals,
- a list of threats together with the description of places in the system where adverse tampering with the information flow can occur,
- the level of protection required or amount of power (in term of accessible computing resources) that is expected from an attacker (or adversary), and
- the projected life span of the system.

Cryptography provides our designer with tools to implement the information protection requested or, in other words, to achieve the security goals expected. The collection of basic tools includes encryption algorithms, authentication codes, one-way functions, hashing functions, secret sharing schemes, signature schemes, pseudorandom bit generators, zero-knowledge proof systems, etc. From these elementary tools, it is possible to create more complex tools and services, such as threshold encryption algorithms, authentication protocols, key establishment protocols, and a variety of application-oriented protocols, including electronic payment systems, electronic election, and electronic commerce protocols. Each tool is characterized by its security specification which usually indicates the recommended configuration and its strength against specific threats, such as eavesdropping and illegal modification of information. The designer can use all the tools provided by cryptography to combine them into a single solution. Finally, the designer has to verify the quality of the solution, including a careful analysis of the overall security achieved.

The second part of cryptology is *cryptanalysis*. Cryptanalysis uses mathematical methods to prove that the design (an implementation of information protection) does not achieve a security goal or that it cannot withstand an attack from the list of threats given in the security specification of the design. This may be possible if the claimed security parameters are grossly overestimated or, more often, if the interrelations among different threats are not well understood.

An attentive reader could argue that cryptography includes cryptanalysis as the designer always applies some sort of analysis of the information protection achieved. To clarify this point, note that the aim of cryptography is the design of new (hopefully) secure algorithms, protocols, systems, schemes, and services, while cryptanalysis concentrates on finding new *attacks*. Attacks (which are a part of cryptanalysis) are translated into the so-called *design criteria* or *design properties* (which are a part of cryptography). The design criteria obtained from an attack allow us to design a system that is immune against the attack.

Cryptography tries to prove that the obtained designs are secure, using all available knowledge about possible attacks. Cryptanalysis carefully examines possible and realistic threats to find new attacks and to prove that the designs are not secure (are breakable). In general, it is impossible to prove that information protection designs are unbreakable, while the opposite is possible – it is enough to show an attack.

1.2 Terminology

Cryptography has developed an extensive vocabulary. More complex terms will be introduced gradually throughout the book. However, a collection of basic terms is presented below.

There is a list of basic security requirements. This list includes: secrecy (or confidentiality), authenticity, integrity, and nonrepudiation. *Secrecy* ensures that information flow between the sender and the receiver is unintelligible to outsiders. It protects information against threats based on eavesdropping. *Authenticity* allows the receiver of messages to determine the true identity of the sender. It guards messages against impersonation, substitution, or spoofing. *Integrity* enables the receiver to verify whether the message has been tampered with by outsiders while in transit via an insecure channel. It ensures that any modification of the stream of messages will be detected. Any modification that results from changing the order of transmitted messages, deleting some parts of messages, or replaying old messages will be detected. *Nonrepudiation* prevents the sender of a message from claiming that they have not sent the message.

Encryption was the first cryptographic operation used to ensure secrecy or confidentiality of information transmitted across an insecure communication channel. The encryption operation takes a piece of information (also called the message, message block, or plaintext) and translates it into a cryptogram (ciphertext or codeword) using a secret cryptographic key. *Decryption* is the reverse operation to encryption. The receiver who holds the correct secret key can recover the message (plaintext) from the cryptogram (ciphertext).

The step-by-step description of encryption (or decryption) is called the *encryption algorithm* (or *decryption algorithm*). If there is no need to distinguish encryption from decryption, we are going to call them collectively *ciphers*, *cryptoalgorithms*, or *cryptosystems.*

Private-key or symmetric cryptosystems use the same secret key for encryption and decryption. More precisely, the encryption and decryption keys do not need to be identical – the knowledge of one of them suffices to find the other (both keys must be kept secret).

Public-key or asymmetric cryptosystems use different keys for encryption and decryption. The knowledge of one key does not compromise the other.

Hashing is a cryptographic operation that generates a relatively short *digest* for a message of arbitrary length. Hashing algorithms are required to be *collision-resistant*, i.e. it is “difficult” to find two different messages with the same digest.

One-way functions are functions for which it is easy to compute their values from their arguments but it is difficult to reverse them, i.e. to find their arguments knowing their values.

The *electronic signature* is a public and relatively short string of characters (or bits) that can be used to verify the authorship of an electronic document (a message of arbitrary length) by anybody.

Secret sharing is the method of distribution of a secret amongst participants so that every large enough subset of participants is able to recover collectively the secret by pooling their *shares*. The class of all such subsets is called the *access structure*. The secret sharing is set up by the so-called *dealer* who, for the given secret, generates all shares and delivers them to all participants. The recalculation of the secret is done by the so-called *combiner* to whom all collaborating participants entrust their shares. Any participant or any collection of participants outside the access structure is not able to find out the secret.

Cryptanalysis has its own terminology as well. In general, cryptographic designs can be either unconditionally or conditionally secure. An *unconditionally secure* design is immune against any attacker with an *unlimited* computational power. For a *conditionally secure* design, its security depends on the difficulty of reversing the underlying cryptographic problem. At best, the design can be only as strong as the underlying cryptographic problem.

An *attack* is an efficient algorithm that, for a given cryptographic design, enables some protected elements of the design to be computed “substantially” quicker than specified by the designer. Some other attacks may not contradict the security specification and may concentrate on finding overlooked and realistic threats for which the design fails. Any encryption algorithm that uses secret keys can be subject to an *exhaustive search attack*. The attacker finds the secret key by trying all possible keys.

Encryption algorithms can be analyzed using the following typical attacks:

- *Ciphertext-only attack* – the cryptanalyst knows the encrypted messages (cryptograms) only. The task is to either find the cryptographic key applied or decrypt one or more cryptograms.

- *Known-plaintext attack* – the adversary has access to a collection of pairs (message and the corresponding cryptogram) and wants to determine the key or decrypt some new cryptograms not included in the collection.
- *Chosen-plaintext attack* – this is the known-plaintext attack for which the cryptanalyst can choose messages and read the corresponding cryptograms.
- *Chosen-ciphertext attack* – the enemy can select her own cryptograms and observe the corresponding messages for them. The aim of the enemy is to find out the secret key. In private-key encryption, the attacker may encrypt new messages into valid cryptograms. In public-key encryption, she may create valid cryptograms from the sample of observed valid cryptograms (*lunchtime attack*).

Authentication algorithms allow senders to incorporate their identity to transmitted messages so each receiver can verify the owner (or sender) of the message. Authentication algorithms can be evaluated using their resistance against the following attacks:

- *Impersonation attack* – the cryptanalyst knows the authentication algorithm and wants to construct a valid cryptogram for a false message, or determine the key (the encoding rule).
- *Substitution attack* – the enemy cryptanalyst intercepts a cryptogram and replaces it with another cryptogram (for a false message).
- *Spoofing attack* – the adversary knows a collection of different cryptograms (codewords) and plans to either work out the key (encoding rule) applied or compute a valid cryptogram for a chosen false message.

Attacks on cryptographic hashing are efficient algorithms that allow a *collision*, i.e. two different messages with the same digest, to be found. All hashing algorithms are susceptible to the so-called *birthday attack*. A weaker form of attack on hashing produces *pseudocollisions*, i.e. collisions with specific restrictions imposed, usually on the initial vectors.

Secret sharing is analyzed by measuring the difficulty of retrieving the secret by either an outsider or an unauthorized group of participants. More sophisticated attacks could be launched by a cheating participant who sends a false share to the combiner. After the reconstruction of the invalid secret (which is communicated to all collaborating participants), the cheater tries to compute the valid one.

1.3 Historical Perspective

The beginnings of cryptography can be traced back to ancient times. Almost all ancient civilisations developed some kind of cryptography. The only exception was ancient China. This could be attributed to the Chinese complex ideogram alphabet – writing down the message made it private, as few could read. In ancient Egypt secret writing was used in inscriptions on sarcophaguses to increase the mystery of the place. Ancient India used its allusive languages to create a sort of impromptu cryptography. Kahn [264] gives an exciting insight into secret communications from ancient to modern times.

Steganography, or secret writing, was probably the first widely used method for secure communication in a hostile environment. The secret text was hidden on a piece of paper by using a variety of techniques. These techniques included the application of invisible ink, masking of the secret text inside an inconspicuous text, and so forth. This method of secure communication was rather weak if the document found its way to an attacker who was an expert in steganography. Cryptography in its early years very much resembled secret writing – the well-known Caesar cipher is an excellent example of concealment by ignorance. This cipher was used to encrypt military orders which were later delivered by trusted messengers. This time the ciphertext was not hidden but characters were transformed using a very simple substitution. It was reasonable to assume that the cipher was "strong" enough as most of the potential attackers were illiterate and it was hoped that the others thought that the document was written in an unknown foreign language.

It was quickly realized that the assumption about an ignorant attacker was not realistic. Most early European ciphers were designed to withstand attacks from educated opponents who knew the encryption process but did not know the secret cryptographic key. Additionally, it was necessary that encryption and decryption processes could be done quickly, usually by hand or with the aid of mechanical devices such as the cipher disk invented by Leon Battista Alberti[1].

At the beginning of the nineteenth century the first mechanical-electrical machines were introduced for "fast" encryption. This was the first breakthrough

[1] Leon Battista Alberti (1404–1472) was born in Genoa, Italy. He was a humanist, architect and principal founder of Renaissance art theory. Alberti is also called the Father of Western Cryptology because of his contributions to the field [264].

in cryptography. Cryptographic operations (in this case encryption and decryption) could be done automatically with minimal operator involvement.

Cipher machines could handle relatively large volumes of data. The German ENIGMA and Japanese PURPLE are examples of cipher machines. They were used to protect military and diplomatic information.

The basic three-wheel ENIGMA was broken by Marian Rejewski, Jerzy Różycki, and Henryk Zygalski, a team of three Polish mathematicians. Their attack exploited weaknesses in the operating procedure used by the sender to communicate the settings of the machine rotors to the receiver [16, 96]. The British team of Alan Turing at Bletchley Park perfected the attack and broke the strengthened versions of ENIGMA. Churchhouse in [97] describes the cryptanalysis of the four-wheel ENIGMA. These remarkable feats were possible due to careful analysis of the cryptographic algorithms, the predictable selection of the cipher machine parameters (poor operational procedures), and significant improvements in computational power. Cryptanalysis was first supported by application of the so-called crypto bombs, copies of the original cipher machines used to test some of the possible initial settings. Later cryptanalysts applied the early computers to speed up computations.

The advent of computers gave both the designers and cryptanalysts new powerful tools for fast computations. New cryptographic algorithms were designed and new attacks were developed to break them. The new impetus in cryptology was not given by new designing tools but rather by new emerging applications of computers and new requirements for the protection of information. Distributed computation and sharing information in computer networks were among those new applications that demonstrated, sometimes very dramatically, the necessity of providing tools for reliable and secure information delivery. Recent progress in Internet applications illustrates the fact that new services can be put on the Net only after a careful analysis of their security features. Secrecy is no longer the most important security issue. In the network environment, the authenticity of messages and the correct identification of users have become the two most important requirements.

The scope of cryptology has increased dramatically. It is now seen as the field that provides the theory and a practical guide for the design and analysis of cryptographic tools, which can then be used to build up complex secure protocols and services. The secrecy part of the field, traditionally concentrated around the design of new encryption algorithms, was enriched by the addition of authentication, cryptographic hashing, digital signatures, secret sharing schemes, and zero-knowledge protocols.

1.4 Modern Cryptography

In 1949 Claude Shannon [467] laid the theoretical foundations of modern private-key cryptography in his seminal work by using information theory to analyze ciphers. He defined the *unicity distance* in order to characterize the strength of a cipher against an opponent with unlimited computational power. He also considered the so-called *product ciphers*. Product ciphers use small substitution boxes connected to larger permutation boxes. Substitution boxes (also called *S-boxes*) are controlled by a relatively short cryptographic key. They provide confusion (because of the unknown secret key). Permutation boxes (*P-boxes*) have no key – their structure is fixed and they provide diffusion. Product ciphers are also termed *substitution-permutation* or *S-P networks*. As the decryption process applies the inverses of S-boxes and P-boxes in the reverse order, decryption in general cannot be implemented using the encryption routine. This is expensive in terms of both hardware and software.

In the early 1970s, Feistel [168] used the S-P network concept to design the Lucifer encryption algorithm. It encrypts 128-bit messages into 128-bit cryptograms using a 128-bit cryptographic key. The designer of the Lucifer algorithm was able to modify the S-P network in such a way that both the encryption and decryption algorithms could be implemented by a single program or a piece of hardware. Encryption (or decryption) is done in 16 iterations (also called rounds). Each round acts on a 128-bit input (L_i, R_i) and generates a 128-bit output (L_{i+1}, R_{i+1}) using a 64-bit partial key k_i. A single round can be described as

$$\begin{aligned} R_{i+1} &= L_i \oplus f(k_i, R_i) \\ L_{i+1} &= R_i \end{aligned} \tag{1.1}$$

where L_i and R_i are 64-bit long sequences, $f(k_i, R_i)$ is a cryptographic function (also called the round function) which represents a simple S-P network. In the literature, the transformation defined by (1.1) is referred to as the Feistel permutation. Note that a round in the Lucifer algorithm is always a permutation regardless of the form of the function $f()$. Also the inverse of a round can use the original round routine with the swapped input halves. The strength of the Lucifer algorithm directly relates to the strength of the cryptographic function $f()$. Another interesting observation is that the design of a Lucifer-type cryptosystem is equivalent to the design of its $f()$ function, which operates on shorter sequences.

The *Data Encryption Standard* (DES) was developed from Lucifer [383] and very soon became a standard for encryption in banking and other non-military applications. It uses the same Feistel structure with shorter 64-bit message/cryptogram blocks and a shorter 64-bit key. As a matter of fact, the key contains 56 independent and 8 parity-check bits. Due to its wide utilization, DES has been extensively investigated and analyzed. The differential cryptanalysis invented by Biham and Shamir [33] was first applied to the DES. Also the linear cryptanalysis by Matsui [320, 321] was tested on the DES.

The analysis of the DES gave a valuable insight into the design properties of cryptographic algorithms. Successors of the DES, whose structures were based on Feistel permutation, include the Fast Encryption Algorithm (FEAL) [347], the International Data Encryption Algorithm (IDEA) [294], and some algorithms submitted as candidates for the Advanced Encryption Standard (AES) and the New European Schemes for Signatures, Integrity, and Encryption (NESSIE) project.

On October 2, 2000 the US National Institute of Standards and Technology announced the winner of their AES competition, the Rijndael algorithm from a team in Belgium. Unlike the DES algorithm that uses the Feistel permutation, Rijndael is based on the general S-P network and needs two separate algorithms, one for encryption and the other for decryption.

Cryptographic hashing has become an important component of cryptographic primitives, especially in the context of efficient generation of digital signatures. MD4 [429] and its extended version MD5 [430] are examples of the design that combines Feistel structure with C language bitwise operations for fast hashing. Although both MD4 and MD5 have been shown to have security weaknesses, their design principles seem to be sound and can be used to develop more secure hashing algorithms.

Both private-key encryption and hashing algorithms can be designed using one-way functions. These constructions are conditionally secure as the security of the algorithms depends upon the difficulty of reversing the underlying one-way functions. In 1976 Diffie and Hellman in their visionary paper [152] introduced a class of trapdoor one-way functions that are easy to invert with the help of some extra information. They showed that such functions can be used to design public-key cryptosystems. Soon after, in 1978, two practical implementations of public-key cryptosystems were published. Rivest, Shamir, and Adleman [428] based their algorithm (the RSA system) on factorization and discrete logarithms. Merkle and Hellman [339] used the knapsack function. Unfortunately, the Merkle-Hellman cryptosystem was broken six years later.

Conventional cryptographic algorithms have a limited lifetime – an algorithm "dies" if an exhaustive attack[2] becomes possible due to progress in computing technology. Conditionally secure cryptographic algorithms are insensitive to the increment of computational power of the attacker. It is enough to select larger security parameters for the algorithm and be sure that the algorithm is still secure.

The design and analysis of conditionally secure cryptographic algorithms is related to easy and difficult problems defined in Complexity Theory. A cryptographic algorithm can be applied in two ways: legally by an authorized user, or illegally by an adversary. Naturally, the authorized use of the cryptographic algorithm should be as efficient as possible, while the illegal handling ought to be difficult. This brings us to the notion of provable security, where breaking a cryptographic algorithm is equivalent to finding an efficient algorithm for solving an intractable problem (such as a factoring, discrete logarithm, or knapsack problem, or any other problem believed to be intractable).

[2] In the case of encryption algorithms, this means that the secret key space can be exhaustively searched. In the case of hashing algorithms, this means that the birthday attack becomes viable.