**The Book Review Column**[1]

by William Gasarch
Department of Computer Science
University of Maryland at College Park
College Park, MD, 20742
email: gasarch@cs.umd.edu

In this column we review the following books.

1. **Introduction to Cryptography** by Johannes A. Buchmann. Reviewed by Andrew Lee. This book is an undergraduate textbook in cryptography which does not assume much prior math background.

2. **Coding Theory and Cryptography: The Essentials, Second Edition** by D.R. Hankerson, D.G. Hoffman, D.A. Leonard, C.C. Lindner, K.T. Phelps, C.A. Rodger, J.R. Wall. Review by Robert J. Irwin. The book tries to cover Coding Theory and Cryptography, two topics with large overlapping audiences that, nevertheless, are usually in different texts.

3. **Cryptography: Theory and Practice** by Douglas R. Stinson. Reviewed by William M. Springer II. This book is an undergraduate textbook in cryptography which does assumes mathematical maturity in the reader.

4. **Foundations of Cryptography: Basic Tools** by O. Goldreich Review by Riccardo Pucella. See next item for brief word on the content.

5. **Modelling and Analysis of Security Protocols** by P. Ryan and S. Schneider Review by Riccardo Pucella. The review is actually a joint review of this book and *Foundations* . . .. The reviewer states that the two books give two different views which complement each other. *Foundations* . . . looks at particular protocols. **Modelling** . . . looks at how you may use protocols with certain properties to build a secure system.

6. **Modern Cryptography, Probabilistic Proofs and Pseudorandomness (Algorithms and Combinatorics, Vol 17)** by Oded Goldreich. Review by Andree Lee. This is a collection of essays about cryptography, probabilistic proofs, and psuedorandomness. The essays are strong on concepts and weak on technical details.

**Books I want Reviewed**

If you want a FREE copy of one of these books in exchange for a review, then email me at gasarchcs.umd.edu
Reviews need to be in LaTeX, LaTeX2e, or Plaintext.

**Books on Algorithms, Combinatorics, and Related Fields**

1. *Algorithms: Design Techniques and Analysis* by Alsuwaiyel.

2. *Immunocomputing: Principles and Appliations* by Tarakanov, Skormin, Sokolova.

3. *Diophantine Equations and Power Integral Bases* by Gaal.

4. *Computational Line Geometry* by Pottmann and Wallner.

5. *The Design and Analysis of Algorithms* by Levitin.

**Books on Cryptography and Books on Learning**

1. *RSA and Public-Key Cryptography* by Richard Mollin.
2. *Data Privacy and Security* by David Salomon.
3. *Elliptic Curves: Number Theory and Cryptography* by Larry Washington.
4. *Block Error-Correcting Codes: A Computational Primer* by Xambo-Descamps.
5. *Logic for Learning* by Lloyd.

**Books on Complexity and Logic**

1. *Essentials of Constraint Programming* by Fruhwirth and Abdennadher
2. *Term Rewriting Systems* by Terese

**Review of**[2]
**Introduction to Cryptography**
**by Johannes A. Buchmann**

Springer Verlag, January 2001
ISBN 0387950346

Reviewer: Andrew C. Lee CS Dept, U. of Louisiana at Lafayette

# 1 Overview

Buchmann's book is a text on cryptography intended to be used at the undergraduate level. As stated in its preface, the intended audiences of this book are "readers who want to learn about modern cryptographic algorithms and their mathematical foundations but who do not have the necessary mathematical background". In less than 300 pages, it gives a concise presentation of many basic methods in modern cryptography. Background knowledge from algebra and elementary number theory are reviewed in the first two chapters. Topics such as elliptic curves cryptography, identification and Public Key Infrastructures are touched upon briefly in the final chapters.

# 2 Summary of Content

The major contents of Buchmann's book can be divided into the several categories. They are summarized as follows:

**Mathematical Preliminaries** It includes carefully selected materials from elementary number theory and algebra. The presentation emphasizes algorithmic techniques. For example, an analysis of the extended euclidean algorithm is provided. Probability are introduced in chapter four, together with Shannon's theorem on perfect secrecy.

[2] 

**Encryption** Substantial amount of materials are devoted to block ciphers. It explains how it can be used in encryption and describes different modes of operations of a block cipher. Classical examples of affine linear block ciphers are provided. It then demonstrates the insecurity of these ciphers by providing a cryptanalysis of affine linear block ciphers. Chapter 5 is devoted to DES (Data Encryption Standard), where the underlying structure of DES are provided in quite details. A brief discussion on DES security is also given.

**Public Key Cryptography** Standard materials in this area are introduced in chapter 7 (e.g. RSA, ElGamal and Rabin encryption; Diffie Hellman Key Exchange etc.). Motivations on the use of public key cryptography are first presented before the introduction of these standard topics.

**Primality Testing and Factoring Algorithms** For primality testing, it includes the discussions of Fermat Test, Carmichael numbers and Miller-Rabin Test. For factoring, its major focus is on quadratic sieve. An analysis of quadratic sieve is also outlined.

**Discrete Logarithms** Chapter 9 discuss the discrete logarithm problem and its algorithmic solutions. It includes both the generic methods (e.g. Shanks baby step giant step method; Pollard's rho algorithm etc.) and specific ones (e.g. index calculus).

**Digital Signatures** Cryptographic hash functions and Message Authentication Code are first introduced. The standard materials regarding digital signatures (e.g. RSA Signatures, ElGamal Signatures and DSA) are then covered.

**Other Topics** Topics such as elliptic curves, identification and public key infrastructure are only covered briefly.

# 3 Opinion

I enjoy reading this book. Despite its brevity, I find the flow of ideas in the book quite clear and the mathematics treatment are rigorous. You can easily find the background motivations and they are usually explained in layman terms. This book also maintain its main theme well. In short, it's focus is in number theoretic algorithms used for cryptographic applications. Readers will find a good exposition of the techniques used in developing and analyzing these algorithms. Readers can also find solutions to selected problems in the appendix. These make Buchmann's text an excellent choice for self study or as a text for students with sufficient mathematics background in elementary number theory and algebra. Note that the language of complexity theory are seldom used in this text.

To keep this book short, omissions seems to be unavoidable. For example, the coverage on notions such as computational indistinguishability and elliptic curves cryptography are minimal. Before ending this review, I would list some suggestions regarding the choice of contents from a readers point of view:

**Security goals** Before going into various topics, a general question that a novice reader like myself may ask is "What are the goals when developing these algorithms ?". The meaning of *security* may have different meanings to different people at different times. Before introducing the main topics, it will be nice to have a general discussion on these basic issues, stating the models that will be examined in this text, the assumptions being made with respective to these models and the limitations (if any) that they may have.

**LIDIA** As many of the algorithms presented are provided via LIDIA (a C++ library developed by the author and his colleagues), one may wonder if LIDIA can be used as a tool to illustrate ideas. While I am visiting the associated website, I learned that LIDIA contains many interesting packages. For example, one may use it to generate elliptic curves that are cryptographically strong. One may also find various algorithms for lattice basis reduction and many other interesting materials. It will be nice if we can use LIDIA to help our understanding of these advanced concepts.

**Background materials** Cryptography is a field that uses many advance mathematics and has immediate applications in real life. In the text the author introduces these connections by citing examples such as the use of smart cards. If possible, materials that describe these developments can be collected, organized and presented to the readers as further references. This perhaps will help the readers, especially the younger generations to recognize the relevance of mathematics in the cutting edge technology.

Review of

**CODING THEORY AND CRYPTOGRAPHY: The Essentials, Second Edition, Revised and Expanded** [3]

**Authors: D.R. Hankerson, D.G. Hoffman, D.A. Leonard, C.C. Lindner, K.T. Phelps, C.A. Rodger, J.R. Wall**

**Series: Pure and Applied Mathematics, Volume 234**

**Marcel Dekker, 2000**

**Hardcover, x + 350 pages, $85.00 from the publisher**

Reviewer: Robert J. Irwin

# 1 Overview

It is pretty much taken for granted that data storage and communication are *reliable*. Increasingly, we expect — or hope — that our recorded and transmitted data are *secure*. The theories of coding and cryptography attend to data reliability and security, respectively. Students interested in coding theory are likely also to be interested in cryptography, and vice-versa. Moreover, there is considerable overlap in the mathematical background required to study the two subjects. Given these strong pragmatic ties, it would be a great boon if one could buy a single text suitable for learning both subjects and still get change back from one's $100 bill.

The book under review is a new and expanded edition of an introductory coding theory text by six of the seven above-named authors [1], all of Auburn University at the time the text was written. The biggest change from the previous edition is the addition of an introduction to cryptography following the text's more extensive coverage of coding theory. Other recent works have covered coding theory and cryptography in one volume, too [4, 5]. I'll say more about them later.

# 2 Summary of Contents

The book is divided into two parts. Part I: Coding Theory comprises the bulk of the text, nine of its twelve chapters. This part has been used to provide material for a two-semester sequence in coding theory

at Auburn for students having at least a "rather elementary knowledge of linear algebra.". Linear block codes predominate, as one would expect, though an entire chapter is devoted to convolutional codes. Part II: Cryptography presents a short introductory course in that subject. The septumvirate of authors wrote the second part for a "diverse audience of graduate and undergraduate students from computer science, engineering, education and mathematics, some of whom will have had only an introductory course in algebra or number theory at the sophomore level." This is a mighty broad readership to serve, one that reflects burgeoning general interest in matters cryptological.

## 2.1 Part I: Coding Theory

As the subtitle indicates, coverage really does stick to the basics. Most of the required mathematics is administered in small doses, just before being applied, so as not to overwhelm weaker hosts. Unutilized mathematical generality is avoided.

The clearly written introductory chapter presents the paradigmatic communication-over-noisy-channel schema and provides some basic information theoretic definitions (pithy quote: "The most important part of the diagram, as far as we are concerned, is the noise, for without it there would be no need for the theory."). The idea of encoding a message to permit error detection is introduced and the maximum likelihood method (MLD) advanced for decoding with error correction. MLD is then analyzed for reliability on a few simple codes chosen to yield mixed results, thus illustrating and prompting further discussion of criteria for code selection.

The first chapter also establishes the loose definition-theorem-proof format used throughout the text. Most sections begin with a few key definitions which are immediately followed by examples. Similarly, examples and exercises accompany most theorems and algorithms, so that students see codes in action straightaway. Theorems, proofs, algorithms and examples are clearly marked. Definitions are not similarly distinguished, though first uses of new terms are italicized in the narrative and referenced in the index. Overall, the presentation style is relaxed and informal, occasionally enlivened by brief chatty interludes. The text is not at all dry.

After the deft set-up, the authors go to work presenting various families of codes, beginning with a chapter introducing simple linear codes, and the vector space concepts needed to understand their properties. Then Hamming bounds and perfect codes are explored, including Hamming and Golay codes. Extended Golay and Reed-Muller codes are also discussed. Cyclic linear codes are next, following a brief review of polynomials over fields of characteristic 2.

Other important code families are studied, to wit: BCH (Bose-Chaudhuri-Hocquenham; this family includes the Reed-Solomon codes, which are covered in a separate chapter), Burst Error-Correcting, Reed-Muller and Preparata. Occasionally, practical applications of particular code families are mentioned to hold interest. Additional facts about finite fields and polynomials are provided when and as needed — often without proof, however, as by way of review. Combinations of encodings are also discussed, such as the use of Reed-Solomon with convolutional codes for space communications, or using burst error-correcting methods in conjunction with other codes.

Over 300 exercises are provided, many of which offer the kind of drill undergraduates need to test their understanding. Full or partial solutions to almost half of them are given in an appendix.

Overall, the narrative and examples of Part I, such as the extended example of Reed-Solomon-based compact disc encoding, present the highlights of the subject neatly, and the reader is usually alerted to simplifications made and real-world details omitted. Uniformity of tone and structure are maintained within this part remarkably well for a work with seven authors.

### 2.2 Part II: Cryptography

Cryptography is covered in three chapters spanning 80 pages altogether. Right away, one notices that Part II is distinct from Part I, aside from subject matter. For example, only the cryptography chapters offer footnotes and end-of-chapter remarks. The edifying and amusing footnotes provide intriguing historical tidbits, often about cryptological embarrassments, that give the reader a satisfying "clued-in" feeling. Chapter end notes provide a useful guide to the bibliography — over three quarters of its 105 entries concern cryptography.

"Classical Cryptography," the lead-off chapter, briskly defines the field and its major application areas: confidentiality, message and sender authentication, message integrity, and non-repudiation. Here, the general communication-with-encryption scenario is limned, the basic vocabulary used in the sequel provided, and a sequence of secret- or symmetric-key encryption schemes is presented. The usual path is taken, starting with simple substitution ciphers and progressing through polyalphabetic block ciphers (the Vigenère cipher) and stream ciphers (the Vernam, or one-time pad cipher), and ending with Feistel ciphers and the Data Encryption Standard (DES). The different flavors of security (unconditional, computational and provable) are informally discussed and Kerckhoffs's principles for selecting ciphers are considered. The one-time pad cipher is given as an example of an unconditionally secure encryption system per Shannon's criterion, which is not rigorously defined, though a reference is provided.

In contrast to the "just in time" approach of the first part, most of the cryptological mathematics expected to be unfamiliar to readers is introduced in a dedicated chapter, "Topics in Algebra and Number Theory." Here, the groundwork is laid for public-key cryptography in 25 pages unrelievedly devoted to higher arithmetic. Topics include the integers modulo $n$, quadratic residues, primality testing, factoring and square roots, and discrete logarithms. Complexity matters are briefly, and informally, addressed.

The final chapter, "Public-key Cryptography," introduces readers to asymmetric encryption schemes. After preliminaries on one-way/trapdoor functions and hashes, the RSA cryptosystem is introduced. Careful mention is made of the fact that, while RSA is based on the difficulty of factoring, it is not known to be equally difficult, nor is factoring itself known to be intractable (intractability is not formally defined in the text). Rabin's related public-key scheme is then given and its difficulty is shown to be closely tied to that of factoring. The ElGamal encryption scheme, based on the unproven intractability of the discrete logarithm problem rather than on factoring, comes next. Applications of public-key cryptography to digital signatures and non-repudiation are covered. The text proper ends with a discussion of several cryptographic protocols: Diffie-Hellman key agreement, zero-knowledge proofs, coin-tossing and mental poker.

Around 80 exercises appear in Part II; a good mix of drill and more substantial problems, often with references. As for Part I, almost half are fully or partially solved in an appendix.

## 3 Opinion

The authors have made a fairly good read of the coding theory part, which cannot hope to compete with the inherent cloak-and-dagger cool of the cryptography part. They clearly took pains to make the former subject as painless as possible, if at the expense of fuller disclosure. There is more than a semester's worth of coding theory material here, up to a year assuming the instructor fills in some gaps. Sharper coverage of which codes are best for particular applications would help the engineering and computer science majors for whom the text seems most suited.

While a bit of linear algebra is cited by the authors as the "minimal prerequisite" for undertaking a coding theory course based on their text, students familiar with discrete probability and the algebra of finite fields will be *much* the happier for it; the book is not self-contained with respect to these subjects. Perhaps the doughty, if under-prepared, reader could orient himself from context, but, e.g., the line "...one which utilizes *Galois fields* [italics added] GF$(2^r)$." may appear as though dropped from a helicopter.

Though compact, Part II provides a good selection of essential results in cryptography, in some cases without proof. A short course could be based on this part, but supplemental material would be needed for a full semester course, especially one for graduate students. Too often, the text would have the reader resort to Kahn's famous techno-history [2] or Stinson's popular text [3] for fuller examples and proofs. Combined coding theory/cryptography courses, long or short, could be taught from this text. However, its two parts are so completely independent of one another that an instructor seeking a more unified treatment of these subjects should look elsewhere.

The bibliography, while clearly not intended to be complete, is select. It contains considerably more, and more up-to-date, entries for cryptography than for coding theory. Some cryptography entries refer to good expository material held in on-line reports and special proceedings as well as to standard texts and research papers. As to matters of production, the book's layout is clear and its typography unobtrusive. Such misprints as I detected seemed relatively benign and correctable from context.

I promised further word on the competition. The recent text of Trappe and Washington [4] also covers both coding theory and cryptography, but with the emphasis reversed: 2 chapters comprising about 80 pages are devoted to information and coding theories, with the remainder of this longer text given over to cryptography. Many of the families of error-correcting codes discussed in the text under review are also covered in [4]; convolutional codes, however, are omitted. Trappe and Washington's coverage of cryptography and its applications is much broader than that of Hankerson, et al, including, for example, entire chapters devoted to e-commerce and digital cash, elliptic curves, and quantum cryptography. Overall, [4] is more self-contained mathematically, providing introductory material on finite fields and a brief review of discrete probability (it is less self-contained where vector space theory is concerned, however). The two authors link up coding theory and cryptography via the McEliece cryptosystem, based on the difficulty of finding the nearest codeword for a linear binary code.

Another book, not quite so recent, by Dominic Welsh [5] looks very promising (at the time of this writing I have not finished with it). This introductory text addresses more or less the same audience as those of Hankerson, et al, and Trappe and Washington, but it is more rigorous than either, and so better suited for graduate students or well-disciplined undergraduates, including mathematics majors. At 257 pages, this is the shortest of the texts mentioned, but by dividing coverage about equally between codes and cryptography, it includes more material on information and coding theory than [4], and more material on cryptography than the book under review. Welsh is mathematically forthcoming; e.g., information theory and complexity issues receive formal treatment. In fact, his text seems to be more of an information theory and cryptography book that includes a substantial amount of material on error-correcting codes. Shannon's approach to information theory, coding theory and cryptography is adopted from the outset, so this text provides a more unified treatment than the others. Given its length and balance, it may be an excellent choice for a one-quarter/semester combined course in coding theory and cryptography.

## References


[1] D.G. Hoffman, D.A. Leonard, C.C. Lindner, K.T. Phelps, C.A. Rodger, J.R. Wall. *Coding Theory: The Essentials*. Marcel Dekker, 1991.

[2] David Kahn. *The Codebreakers: The Story of Secret Writing*. revised edition, Scribner, 1996

[3] Douglas Stinson. *Cryptography: Theory and Practice*. CRC Press, 1995 (second edition, 2002)

[4] Wade Trappe and Lawrence C. Washington. *Introduction to Cryptography with Coding Theory*. Prentice-Hall, 2002.

[5] Dominic Welsh. *Codes and Cryptography*. Oxford U. Press, 1997.

# Review of
# Cryptography: Theory and Practice, Second Edition
# Author of Book: Douglas R. Stinson
# Publisher: CRC Press (339 pages)

William M. Springer II( wmspringer@acm.org)[§]

October 6, 2003

## 1 Introduction

For thousands of years, men have used codes and ciphers to communicate in secret. Historically, the security of a cryptosystem was based in secrecy: the enemy was unaware of how messages were being encoded, and if there were intercepted, would have no idea how to decrypt them. In modern times, messages are often encrypted on the assumption that the enemy knows everything about the cryptosystem, with the exception of the secret key used to encrypt the message. As a result, modern cryptosystems are often mathematically complex, relying on functions that are difficult to break even when massive computing power is brought to bear. Mathematically defined, a cipher is a function which takes as input a plaintext message $x$ and a key $k$, and returns an encrypted message $y$. A cryptosystem is often defined as a five-tuple $(P, C, K, E, D)$ where $P$ is the set of all possible plaintexts, $C$ is the set of all possible ciphertexts, $K$ is the keyspace, or the set of all possible keys, and $E$ and $D$ are encryption/decryption functions. $E$ and $D$ are chosen such that for every key $k$, $E_k$ maps $P$ onto $C$, $D_k$ maps $C$ onto $P$, and $D_k(E_k(x)) = x$ for every plaintext element $x$ in $P$. These functions can be anything from simple substitution (replace every $A$ with $D$, for example) to complex mathematical functions.

## 2 Chapter 1: Classical Cryptography

No introduction to cryptography would be complete without mentioning some of the classical ciphers, such as the simple Caesar Cipher, which is a simple shift cipher purportedly used by Julius Caesar. For $P = C = K = Z_{26}$, (that is, the plaintext, ciphertext, and keyspace are all integers mod 26, where $A = 0$, $B = 1$, etc) we define $E_k(x) = (x + k) \pmod{26}$ and $D_k(y) = (y - k) \pmod{26}$. Other monoalphabetic ciphers covered in this chapter include the Substitution Cipher (where each letter is represented by another letter; this is known as a permutation cipher) and the Affine Cipher (in which the encryption functions are restricted to the form $e(x) = (ax + b) \pmod{26}$). Polyalphabetic ciphers include the Vigenere cipher, which is a shift cipher that uses a different shift for each letter (for example, a Vigenere Cipher with key "cryptography" would encrypt a message as $(x_1 + 2, x_2 + 17, \ldots)$ and the Hill Cipher, which uses matrix multiplication. This chapter also introduces the reader to Alice and Bob (residents of every cryptography text) and to basic cryptanalysis, including ciphertext only, known plaintext, chosen plaintext, and chosen

ciphertext attacks. This part of the chapter focuses on frequency analysis, in which the known frequencies of letters and digrams in common English sentences are compared with encrypted letters and digrams in the ciphertext, and the Kasiski text, which is used to determine the key length for polyalphabetic ciphers.

## 3 Chapter 2: Shannon's Theory

In cryptography, there are three types of security: computational security, which means that the best algorithm for breaking the cryptosystem requires a very large number of operations; provable security, which means that breaking the cryptosystem is at least as hard as solving some other difficult problem, and unconditional security, where the cryptosystem can never be broken even with infinite computational resources.

Proving the security of a cryptosystem involves basic probability theory; for example, if a cryptosystem is unconditionally secure, then the probability that a message is $x$, given the encrypted message $y$, is the same as the probability that the message is $x$; that is, knowing $y$ gives you no information about the original message. Shannon gave cryptography the tool of entropy, which in this context is a measure of information or uncertainty. The example used here is the flip of a fair coin; the coin can land either heads or tails with equal probability. As we can encode heads with a 1 and tails with a 0, the information (or entropy) of a coin toss is one bit. Similarly, the entropy of n coin tosses is n bits, as we can encode the n tosses with a string of length n.

## 4 Chapter 3: Block Ciphers and the Advanced Encryption Standard

Introduced briefly at the end of chapter 2, product ciphers encrypt a message normally, then encrypt it again using a different key or method. The AES cipher, which is now the official standard for encryption, is one such cryptosystem. AES, which accepts keys of length 128, 192, or 26 bits, and breaks the message into blocks of 128 bits, goes through a variable number of rounds depending on the length of the key. If the key is 128 bits, 10 rounds are required, increasing to 12 for 192 bits and 14 for 256 bits. In each round, a variety of operations (mainly row and column shifts) are performed, thoroughly scrambling the original message. A new and widely tested cryptosystem (AES was originally Rijndael, one of 15 AES candidates accepted by the NIST (National Bureau of Standards, now the National Institute of Science and Technology) and went through three years of inspection and testing before being accepted in late 2001), AES is secure against all known attacks, meaning that there are no attacks known which are significantly faster than an exhaustive search of the keyspace. This chapter covers the old Data Encryption Standard (DES), AES, Linear Cryptanalysis, and Differential Cryptanalysis.

## 5 Chapter 4: Cryptographic Hash Functions

While encrypting data may keep it from being read, the encryption is no guarantee against the data being altered. Hash functions can be used to create an authentication code, or fingerprint, insuring that the message received is the same as the message that was sent. This chapter covers several algorithms for authorization codes and evaluates their security.

## 6 Chapter 5: The RSA Cryptosystem and Factoring Integers

In most cryptosystems, the decryption function is the same as the encryption function, or is easily derived from it; such a system is called a symmetric-key cryptosystem. In a symmetric-key cryptosystem,

the communicating parties share a common key, which must be kept secret; this can lead to problems with key distribution. RSA, invented in 1977 by Rivest, Shamir, and Adleman, is an example of a public-key cryptosystem. Public-key encryption relies on so-called one-way functions, where the encryption function is easy to compute from the decryption function, but not the inverse. One commonly used function is factoring large numbers; given two large primes, it is easy to multiply them together, but difficult to find the original numbers given the product. RSA uses this function; two large primes p and q are chosen to be 512-bit primes, making the product a 1024-bit number. This chapter discusses several results from number theory, including the Euclidean Algorithm and the Chinese Remainder Theorem, then discusses the RSA cryptosystem, testing for primes, factoring, and other attacks on RSA.

## 7 Chapter 6: Public-Key Cryptosystems Based on the Discrete Log Problem

As previously mentioned, public key cryptosystems depend on having appropriate one-way mathematical functions; one such is the discrete logarithm problem. The security of these cryptosystems is based on the fact that finding discrete logarithms is generally difficult, while exponentiation is relatively easy. Several cryptosystems built around discrete logarithms are described, including the well-known ElGamal Cryptosystem. This chapter also discusses similar systems based on finite fields and elliptic curves; the end of the chapter covers the security of ElGamal systems and the Diffie-Hellman problems, which are problems related to Diffie-Hellman key agreement protocols.

## 8 Chapter 7: Signature Schemes

In the physical world, documents often require signatures to verify their validity. The same holds true for digital documents, but special problems apply. First, there must be a way to guarantee that the signature is genuine; it must come from the person it purports to belong two. Secondly, the signature must be somehow bound to the document, so that a valid signature cannot be copied onto something entirely different. Finally, there must be a way to prevent reuse; it would hardly do for a digital dollar to be spent multiple times!

Signature schemes are similar to public key encryption, and indeed they work well together. For example, suppose Alice is sending a message to Bob. After writing her message, she computers a signature based on the message (so it cannot be reused for a different message) and encrypts it using her private key; she then encrypts both the message and the signature using Bob's public key. When Bob receives the message, he decrypts it using his private key, then uses Alice's public key to decrypt her signature and verify that she sent the message.

## 9 Conclusion

I felt that the book was quite readable, although the reader will probably want to be familiar with the terminology of group theory before attempting it. The typos occasionally make a section unclear; the beginning reader will want to correct them using the erreta listing at http://www.cacr.math.uwaterloo.ca/ dstinson/CTAP2/errata.html. My main complaint is the price of the book; for nearly $80 you get a relatively short book (barely over 300 pages) which leaves out some important topics such as Quantum Cryptography; those subjects are scheduled to appear in a companion volume.

Joint Review[1] of
**Foundations of Cryptography: Basic Tools**
**Cambridge University Press**
by O. Goldreich
and
**Modelling and Analysis of Security Protocols**
by P. Ryan and S. Schneider
**Addison Wesley**

Review by Riccardo Pucella
Department of Computer Science
Cornell University

# Introduction

There are essentially two schools, advocating two general approaches to the problem of reasoning about the security properties of a system. What do I mean by a security property? Examples of these include secrecy (ensuring that only authorized parties get access to a piece of information), integrity (ensuring that messages exchanged between parties are not modified in transit), and authenticity (ensuring that parties can ascertain the origin of messages). Reasoning about the security of a system basically means figuring out whether the security properties of interest hold in the presence of an adversary that attempts to attack the system by manipulating the environment in which the system executes. For instance, the adversary may intercept, modify, and redirect messages, may pose as different parties, etc. In order to prevent the adversary from successfully attacking the system (and thereby access secret information, or corrupting messages without any party noticing, depending on what the security property is guaranteeing should not happen), the system will typically make use of various encryption schemes to encrypt the messages exchanged by the parties, signature schemes to digitally sign messages, and communication protocols indicating, say, the pattern of message exchanges that need to occur between the parties.

As I said, historically, two schools have emerged concerning the analysis of security in such systems. Very roughly speaking, the first school, which is also the oldest, focuses on the underlying mechanisms for providing secure messaging, such as encryption schemes (for instance, DES, AES, Blowfish, RSA, etc), or signature schemes. The central theme in this approach is one of complexity—how difficult is it to "break" the scheme—and reliance on probabilities. Here, the adversary is assumed to possess a certain amount of computational abilities, without going into the details of what exactly those abilities are. For instance, a common assumption is to take the adversary to be able to perform arbitrary probabilistic polynomial-time computations.

The second school has focused not so much on the properties of the underlying schemes, but rather on the way these schemes are used in communication protocols. The kind of analysis done is more combinatorial in nature. For instance, many protocols have flaws that are independent of the security of the underlying encryption schemes. To put it bluntly, no encryption scheme is secure if you stupidly reveal the key used to encrypt messages during a protocol exchange. This is of course an extreme example, but many security protocols fail due to the misuse of perfectly good encryption schemes. To help concentrate on that aspect of security protocols, the approach is to completely abstract away from the encryption or signature schemes, assuming them to be perfect, and rather than allow the adversary to break the schemes, only allow him to intercept messages, construct new messages from messages he has intercepted, and encrypt and decrypt data for which he has the corresponding key. While this approach may seem less powerful than the preceding version (because the adversary is so limited), it has the distinct advantage of allowing for automatic

verification tools.

I should emphasize that while a priori the focus of the two approaches is different—arguably, the first approach tends to address lower-level questions than the second approach—in recent years there has been an attempt at reconciling these approaches. The difficulties in this project stem from the fact that the approaches represent two completely different philosophies with respect to the meaning of security properties, and exhibit vastly different methodologies and tools. In the interim, let's look at what these two books have to offer us.

## Foundations of Cryptography: Basic Tools

Goldreich's book exemplifies the computational-complexity approach to security and cryptography, what I have called the first school. By and large, this approach focuses on the properties of the building blocks of many essential security mechanisms, such as encryption schemes, or signature schemes. It focuses on an adversary essentially defined by its computational abilities.

**Chapter 1: Introduction.** This chapter gives a very nice overview of the philosophy underlying the computational-complexity approach to cryptography. It also reviews some of the background required for this book, namely probability theory, and computational complexity models (including the probabilistic polynomial time complexity classes, and non-uniform boolean circuits).

**Chapter 2: Computational Difficulty.** This chapter introduces the core construct of much of cryptography, *one-way functions*. Roughly speaking, a one-way function is a function that is efficient to compute in one direction, but difficult to invert. This ties in with complexity theory in the following way. The existence of one-way functions requires the existence of hard (on average) problems. Furthermore, it should be possible to generate hard instances of those problems, along with enough "auxiliary" information to help solve the instance of the problem if one is given that auxiliary information. Different notions of one-way functions are defined (strong, weak, non-uniform), along with candidates. (Note that the existence of one-way functions is a complexity-theoretic assumption, in the same sense that P$\neq$NP is often taken as a complexity-theoretic assumption.) Saying that a function $f$ is one-way basically says that given a value $y$, it is difficult to find the pre-image of $y$ under $f$. However, it may be the case that partial information on the pre-image is easy to compute. To address this issue, the notion of *hard-core predicates* is introduced, which yields more flexibility in specifying the difficulty of inverting one-way functions.

**Chapter 3: Pseudorandom Generators.** This chapter introduces fundamental concepts in the modern theory of cryptography. Roughly speaking, a *pseudorandom generator* is an efficient (polynomial-time) deterministic algorithm that transforms a short randomly chosen string into a much longer "pseudorandom" string. A pseudorandom string is a string that is *computationally indistinguishable* from a true random string by efficient algorithms. One consequence is that for any string generated by a pseudorandom generator, no efficient algorithm can predict the bit following any given prefix of the string. The first part of the chapter defines computational indistinguishability, then pseudorandom generators. The bulk of the chapter investigates the relationship between pseudorandom generators and one-way functions. Pseudorandom generators exist if and only if one-way functions exist. The construction of pseudorandom generators from a special class of one-way functions is presented. (The derivation for arbitrary one-way functions, much more complex, is sketched.) The generalization of pseudorandom generators to pseudorandom functions is also discussed. Intuitively, a pseudorandom function is a function that cannot be distinguished from a truly random function by any efficient procedure that can sample the function at arbitrary points.

**Chapter 4: Zero-Knowledge Proofs.** This chapter (which takes up half the book) presents one of the most remarkable devices in complexity theory, zero-knowledge interactive proof systems. Roughly speaking, the setting is one in which one party $A$ proves an assertion to another party $B$, so that $B$ is convinced of the validity of the assertion, but where $B$ *does not learn anything* beyond the fact that the assertion is true (and its consequences). This has direct applications to security, where different parties may have access to different pieces of information that they may want to share, but without revealing anything beyond the fact that they have that piece of information. Defining zero-knowledge proofs requires introducing the notion of an interactive proof system, a topic with many complexity theoretic applications beyond cryptography. Informally, an interactive proof system consists of two parties, a "prover" whose task is to produce a proof of an assertion, and a "verifier" whose task is to verify the validity of a proof. The parties can interact—for instance, the verifier can interrogate the prover during the verification process. From this, one can define a zero-knowledge interactive proof system, which requires defining the notion of knowledge derived from a proof. Following these definitions, the main result of the chapter is presented: a method for constructing zero-knowledge proofs for every language in NP. In other words, it is possible to construct zero-knowledge interactive proof systems for any given language $L$ in NP, that proves queries about language membership in $L$. For instance, one can construct a zero-knowledge proof system that answers queries as to whether a graph is 3-colorable, in such a way that the verifier is convinced that the graph is indeed 3-colorable, but does not learn anything beyond that, including how to 3-color the graph. This construction relies on one-way functions (in the form of bit commitment protocols). A discussion of the limitations of zero-knowledge proof systems follows. Some further topics explored in this chapter include:

- proofs of knowledge, where the prover not only asserts the existence of some object—such as a 3-coloring of a graph—by also knowledge of that object;
- computationally sound proofs, which relax some of the correctness requirements of zero-knowledge proofs system;
- constant-round zero-knowledge proofs, where one restricts the number of rounds of interaction between the prover and the verifier to a constant;
- non-interactive zero-knowledge proofs, where the prover and verifier do not closely interact. Rather, there is a single message sent from the prover to the verifier. However, both parties have access to a uniformly chosen random string of bits (chosen by a trusted third party, for instance);
- multi-prover zero-knowledge proofs, a generalization of interactive proof systems where a verifier is allowed to interact with multiple provers.

**Opinion.** This is a very complete introduction to the basics of modern complexity-theoretic cryptography. It is a solid foundation for understanding much of the current work in this area. Note that this is the first volume of a planned three-volume series. Used by itself, this volume is possibly more suited for a first or second-year graduate course on complexity theory than a course on straight cryptography. This should improve once volume 2 and 3 are out. Volume 2 (which is outlined in an appendix of this book) will apply the ideas of this book to the basic problems of defining encryption schemes, signature schemes, and more general cryptographic protocols. Note that draft chapters of the second volume are available online.

## Modelling and Analysis of Security Protocols

Ryan and Schneider's book exemplifies the formal methods approach to security, what I have called the second school. Here, the focus is much more on the combinatorial issues surrounding the use of cryptographic primitives. The idea is that once you have cryptography, and you want to securely exchange

information with other parties, then having cryptographic schemes is but a first step. After that, you want to actual *use* these schemes in such a way that you can achieve some higher-level goals, such as mutual authentication, or secret sharing. How can you do that? You need to construct communication protocols. What's interesting is that some of these protocols have problems *irrespectively* of the strength of the cryptographic primitives. More to the point, even if we assume perfect cryptography, there are protocols that can be broken based only on the way the messages are constructed. This is what the formal methods approach takes as a starting point: assume we do have perfect cryptography (so that the adversary will not attempt to "break" the crypto), represent the protocol through one of many notations people have devised for studying communication protocols, and analyze the protocol in the presence of the kind of adversary alluded to above.

**Chapter 0: Introduction.** This chapter is just an introduction to the problem of reasoning about security protocols, of the kind I described earlier. It also includes a discussion of the kind of properties one may be interested in proving about security protocols.

**Chapter 1: An Introduction to CSP.** This chapter introduces CSP, Hoare's Calculus of Sequential Processes, the language in which systems are described in the book. Roughly, the language allows one to define a system as the parallel composition of processes that communicate via shared channels. A key feature of CSP, in the view of the authors, is that the same language used to describe a system can be used to specify properties of that system. Intuitively, a property can be described by giving the "ideal" behavior of the system, by abstracting away the details of the implementation. One can then show that the process representing the implementation of the system "refines" the process representing the property. This notion of refinement is given a formal definition in this chapter, and is central to the whole approach of reasoning about security protocol in CSP.

**Chapter 2: Modelling Security Protocols in CSP.** In this chapter, CSP is put to the use of describing security protocols. The idea is straightforward: put in parallel processes representing the behavior of all the parties of the system, including any trusted servers. Since security protocols typically rely on cryptographic primitives, we need to define in CSP a suitable datatype for representing the values exchanged by the parties. We also need to add a process representing the adversary. Such a process simply captures the fact that the adversary can intercept and redirect messages, compose new messages from old, and apply encryption and decryption operators.

**Chapter 3: Expressing Protocol Goals.** This chapter focuses on the problem of expressing security goals in the CSP notation. The goals covered include secrecy goals (that a given value is never revealed to the adversary), authentication goals (verification of a party's identity), non-repudiation (that no party can deny having sent or received a message), and anonymity (protecting the identity of the parties involved in the protocol). All of these properties are expressed as CSP processes, and a protocol satisfies a property if it refines that property, as described in Chapter 1.

**Chapter 4: Overview of FDR.** This chapter introduces a tool to automatically check that a process refines another. As we have seen, since we represent properties as CSP processes, this gives a way to check that a process meets a specification. The tool is FDR, a commercial tool from Formal Systems (Europe). The underlying mechanisms of the tool are surveyed in the chapter.

**Chapter 5: Casper.** While FDR provides a mechanical way to check that a CSP process satisfies a specification itself written as a CSP process, it is time-consuming and error-prone to directly manipulate the full CSP process describing a particular protocol, and the CSP process corresponding to the particular security

property to verify. To simplify this verification, a tool called Casper is introduced, that takes as input a script representing a protocol as a sequence of message exchanges, as well as a succinct representation of the property to check, and automatically generates the CSP code corresponding to the protocol and the property, suitable for checking via FDR.

**Chapter 6: Encoding Protocols and Intruders for FDR.** This chapter is a careful study of the encoding of the protocols and properties performed by Casper. Many optimizations are performed in order to produce code that can be checked efficiently by FDR. These include the modelling of the deductive system used by the adversary to derive new facts from intercepted messages, as well as the treatment of algebraic equivalences between values exchanged by the different parties.

**Chapter 7: Theorem Proving.** The approach to verifying protocols described in the previous chapters can only deal with systems with finitely many states. This imposes a restriction on, say, the number of parties in a protocol, or the number of simultaneous instances of the protocol being executed in the system. In this chapter, a general proof technique is presented to establish properties even when the system under consideration is not finite. This relies on the notion of a rank function, which can be used as a sort of invariant to be preserved in the analysis of the protocol.

**Chapter 8: Simplifying Transformations.** The methods introduced in the previous chapters work well for small protocols. For dealing with more involved protocols (including most realistic ones), this chapter examines transformations that can be applied to protocols. These transformations have the property that if an attack is possible in the original protocol, this attack will also be possible in the simplified protocol. Therefore, studying the simplified protocol will not make one miss a flaw of the original protocol. The CyberCash Main Sequence protocol is used as a case study.

**Chapter 9: Other Approaches.** This chapter compares the CSP approach to other approaches to analyze security protocols. These include BAN logic, the NRL protocol analyzer, Strand Spaces, Paulson's inductive assertions approach, and the spi-calculus, to name but the most popular.

**Chapter 10: Prospects and Wider Issues.** This chapter concludes the book by describing some issues that remain to be addressed. These include a more realistic treatment of cryptography (hence bringing in complexity-theoretic issues), as well as a discussion of the combinatorial size explosion that occurs when analyzing protocols in the style advocated by the book.

**Opinion.** This introductory book, suitable for an advanced undergraduate course, gives a good overview of a particular way to apply formal methods to study security protocols. This book is not foundational in any sense of the word. Rather, it shows how to analyze protocols given a particular way to represent protocols (CSP) and a particular tool for verifying properties (FDR). However, in the process, it does present most of the issues common to all formal method approaches to security protocol analysis. Perhaps more importantly, it is to the best of my knowledge the only book on the market that addresses in depth the topic of formal methods applied to security protocol analysis.

**Review of**
**Modern Cryptography, Probabilistic Proofs and Pseudorandomness**
**Algorithms and Combinatorics, Vol 17**
**by Oded Goldreich**

Springer Verlag, January 1999
ISBN 354064766X

Reviewer: Andrew C. Lee C.S. Dept, U. of Louisiana at Lafayette

# 1 Overview

Modern cryptography, probabilistic proofs and pseudorandomness are three areas in theoretical computer science that demonstrate the interplay between randomness and computations. In Goldreich's book you will find a collection of survey articles written for each of the above topics. In each survey, materials ranging from basic notions to recent advanced results are presented. The author seeks to provide a clear perspective to the reader of what these areas are. Although they are closely related areas, each article can be read independently. An extensive bibliography and four appendices are included. These appendices provides background knowledge on computation and randomness, examples on randomized computations in other domains, simplified proofs on two basic results and pointers to survey articles written by the author. These surveys are also related to randomness and computation (e.g. Yao's XOR lemma, Levin's Theory of Average-Case Complexity etc.).

This work is NOT a pure formal treatment of these topics. In each survey the author first provides a summary that describes briefly the perspective he has. Materials are then selected and presented to illustrate and support that perspective. Most of the materials are focused on the essentials and technical details are usually kept to a minimum.

# 2 Summary of Content

For each survey chapter we will first cite the author's own summaries and then combine then with the reviewer's observation.

**The Foundations of Modern Cryptography**

> *Author's own summary:* In our opinion, the Foundations of Cryptography are the paradigms, approaches and techniques use to conceptualize, define and provide solutions to natural cryptographic problems. In this chapter we survey some of these paradigms, approaches and techniques as well as some of the fundamental results obtained using them. Special effort is made in attempt to dissolve common misconceptions regarding these paradigms and results.

The Special effort said above mainly refers to the classification of works in this area into two types of activities. They are the *definitional activity* and the *constructive activity*. He further motivates the reader to view results as either a *plausibility result*, *an introduction of paradigms and techniques which may be applicable in practice*, or, *a presentation of schemes which are suitable for practical applications*. The central paradigms being stressed here are computational difficulty, computational

indistinguishability and the simulation paradigm. After explaining the above notions in quite details, the following topics are introduced: Pseudorandomness, Zero-Knowledge, Encryption, Signatures, Cryptographic Protocols. At the end of the survey, suggestions for future reading and research are provided.

**Probabilistic Proof Systems**

> *Author's own summary:* Various types of probabilistic proof systems have played a central role in the development of computer science in the last decade. In this chapter, we concentrate on three such proof systems – interactive proofs, zero-knowledge proofs, and probabilistic checkable proofs – stressing the essential role of randomness in each of them.

Apart from the three main proof systems mentioned above, this survey also contains a brief overview on variants on the basic models of probabilistic proofs. These include Multi-Prover Interactive Proof Systems, Computational-Sound Proof Systems, Refereed Games etc. This survey is concluded by a comparison among various notions, historical remarks and a collection of open problems, showing that there are still important directions for future research in these areas.

Regarding the technical content, you will find sketches of the proofs of some famous theorems, such as **IP=PSPACE**, **NP= PCP(log,O(1))** and many others in this chapter.

**Pseudorandom Generators**

> *Author's own summary:* A fresh view at the question of randomness was taken in the theory of computing: It has been postulated that a distribution is pseudorandom if it cannot be told apart from the uniform distribution by an efficient procedure, The paradigm, originally associating efficient procedures with polynomial-time algorithms, has been applied also with respect to a variety of limited class of such distinguishing procedures. Starting with the general paradigm, we survey the archetypical case of pseudorandom generators (withstanding any polynomial-time distinguisher), as well as generators withstanding space-bounded distinguishers, the derandomization of complexity classes such as BPP, and some special purpose generators.

This chapter begins by introducing three different approaches to deal with randomness. They are Shannon's information-theoretic approach, Kolmogorov's complexity approach and the complexity-theoretic approach. It focuses on the third approach and introduces the generic framework for defining pseudorandom generators. A considerable amount of work is to explain the proof of the derandomization of BPP and the introduction of a number of special purpose generators. It includes Pairwise-Independence Generators, Small Bias Generators, Random walk on expanders etc.. This survey is concluded by summarizing various pseudorandom generators with relevant parameters in the form of a table. Finally it provides a discussion on each conceptual approaches described earlier in this chapter. A list of open problems are also stated.

# 3 Opinion

This book is informative and rich in content. While writing up this review I realize the extensiveness of materials covered in these three surveys. It seems hard to further *compress* these materials any further.

While one cannot expect to find all the details in this book, you will find a solid and structural view by the author on each of these three areas. To the reviewer, the most appealing feature of this book is that it leans toward the intuition and historical motivations around these topics. I believe that it is an excellent resource for students and researchers who has sufficient background in complexity theory (Note: Earlier versions of these materials may had been used in related conference workshops). You can also find a list of errata from the author's website. If you wish to study a related topic (e.g. derandomization of time complexity classes and constant-depth circuits, the IP=PSPACE proofs and etc.) this book will probably give you a good collection of background motivations and nice discussions as well.

If you have decided to use this text for your personal study, I would recommend you to have access to some bibliographic resources (e.g. ACM digital library, Citeseer etc.) while reading the text. It is especially the case when you reach the later subsections of each survey. Many side topics presented there can only be covered or mentioned very briefly. If these topics interest you, you probably will prefer to look at the abstracts of the related papers (at least) during the reading of the text.

One way to utilize this great resource is to use it in a seminar setting (e.g. a graduate student seminar course). Participants may first review the viewpoints from the surveys. After studying and presenting the core materials from the original papers, the participants may discuss, debate, compare, contrast the viewpoints they share and the ones provided in the text. I believe that these materials will provoke many thoughts and meaningful discussions.