BULLETIN (New Series) OF THE
AMERICAN MATHEMATICAL SOCIETY
Volume 41, Number 3, Pages 357–367
S 0273-0979(04)01011-0
Article electronically published on February 24, 2004

RSA and Public-Key Cryptography, by Richard Mollin, Chapman and Hall/CRC, Boca Raton, FL, 2003, xii + 291 pp., $79.95, ISBN 1-58488-338-3

Introduction to Cryptography, by Hans Delfs and Helmut Knebl, Springer-Verlag, Berlin, 2002, xiv + 310 pp., $49.95, ISBN 3-540-42278-1

Cryptography: Theory and Practice, by Douglas Stinson, Chapman and Hall/CRC, Boca Raton, FL, 2002, xxi + 339 pp., $79.95, ISBN 1-58488-206-9

Algebraic Aspects of Cryptography, by Neal Koblitz, Springer-Verlag, Berlin, 1998, ix + 206 pp., $79.95, ISBN 3-540-63446-0

Elliptic Curves: Number Theory and Cryptography, by Lawrence Washington, Chapman and Hall/CRC, Boca Raton, FL, 2003, xi + 428 pp., $79.95, ISBN 1-58488-365-0

Elliptic Curves in Cryptography, by Ian Blake, Gadiel Seroussi, and Nigel Smart, Cambridge University Press, Cambridge, 1999, xv + 204 pp., $42.00, ISBN 0-521-65374-6

Modern Cryptography, Probabilistic Proofs, and Pseudorandomness, by Oded Goldreich, Springer-Verlag, Berlin, 1999, xv + 182 pp., $97.00, ISBN 3-540-64766-X

Foundations of Cryptography: Basic Tools, by Oded Goldreich, Cambridge University Press, Cambridge, 2001, xix + 372 pp., $75.00, ISBN 0-521-79172-3

The Design of Rijndael: AES — the Advanced Encryption Standard, by Joan Daemen and Vincent Rijmen, Springer-Verlag, Berlin, 2002, ix + 238 pp., $44.95, ISBN 3-540-42580-2

Handbook of Applied Cryptography, by Alfred Menezes, Paul van Oorschot, and Scott Vanstone, CRC Press, Boca Raton, FL, 1997, xxviii + 780 pp., $99.95, ISBN 0-8493-8523-7

Whitfield Diffie and Martin Hellman startled the computer security world in 1975 with their paper "New Directions in Cryptography", which introduced public-key cryptography [7]. There is now evidence that three cryptographers at the British Government Communications Headquarters (GCHQ), the British equivalent of the National Security Agency, may have predated this work as well as the discovery of the Rivest-Shamir-Adleman (RSA) public-key encryption algorithm [9]. It appears, however, that the British researchers did not appreciate the significance of their discovery.

Since the mid 1970s, cryptography has become big business, a bestseller (*Applied Cryptography* by Bruce Schneier has sold over two hundred thousand copies), and an extremely active area of research at the intersection of mathematics and computer science. A plethora of new books in the area currently floods the bookstores. This review cannot discuss them all. Instead I will delineate the subareas of this burgeoning field and focus on some of the texts that I believe will be of more interest to *Bulletin* readers.

2000 *Mathematics Subject Classification.* Primary 94A60.

Many mathematicians are familiar with the RSA cryptosystem in which Alice makes public an integer n which is the product of two large primes p and q along with an integer e relatively prime to $\phi(n) = (p-1)(q-1)$ (where $\phi(n)$ is the Euler ϕ-function). Bob can securely send his digitized message $\mathcal{M} < n$ to Alice by the encryption:

$$\mathcal{C} = \mathcal{M}^e \pmod{n};$$

Alice decrypts through

$$\mathcal{C}^d \pmod{n},$$

where the integer d satisfies the equation $ed + \phi(x)y = 1$ (alternatively, $ed \equiv 1 \pmod{\phi(n)}$). The security of the system relies on the fact that factoring large integers (on the order of 2,000 bits, say) is computationally infeasible. Thus, even though n and e are public and the encrypted message, $\mathcal{C}$, travels over the insecure wires of the Internet, the message is secure against eavesdroppers. The message nonetheless presents no difficulty for Alice to decrypt, for Alice knows the factorization of n and can easily, that is, computationally efficiently, calculate the decryption exponent d.

This simple mathematics is what enables the Secure Socket Layer (SSL) communications that secure Web transactions. When you access the check-out page at Amazon, your machine—the client—and the Amazon computer—the server—do a negotiation. The Amazon server "authenticates" itself to your PC, "proving" that it is a genuine Amazon machine. First the two machines determine which algorithm they will use to encrypt your data (credit card number and order). Your PC chooses a *session key,* an encryption key that will be used only to encrypt this Web session and sends this to the Amazon server. But the PC must communicate this information using an unprotected channel: the Internet. The session key is sent RSA encrypted with Amazon's public key, provided to the PC on the certificate the client received from the Amazon server.

The RSA encryption used in the SSL protocol is not quite as simple as the description earlier. That straightforward version "leaks" information. One way is *semantically*: anyone who learns that $\mathcal{C}_1$ and $\mathcal{C}_2$ are the encryptions of $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively, knows that $\mathcal{C}_1\mathcal{C}_2$ is the encryption of $\mathcal{M}_1\mathcal{M}_2$. While this does not appear to be a terrible leak of information, it is nonetheless a leak. Good cryptosystems do not reveal *any* information unnecessarily. Thus the SSL implementation of the RSA algorithm includes some "padding" of the input.

Several years ago a problem was discovered in the SSL protocol. If the plaintext had not been correctly formatted before it was encrypted, the server would respond with an error message. Bleicherbacher showed that a potential attacker, by sending a slew of carefully determined messages, could use the responses to decrypt a targetted message [4], [24], [2]. The problem was corrected by changing the format used to prepare messages for RSA encryption. This type of problem — the actual implementation of cryptographic algorithms in practical systems — is quite important, but for space reasons, I will confine my review to purely mathematical attacks.

If we are to depend upon encryption for securing our on-line communications, our purchases, our ATM transactions, our cell phone conversations, we will need to base security on something more precise than the inexact phrase "factoring large integers is computationally infeasible". We will need definitions, theorems, clarity, and rigor. We will need mathematics.

Let us begin. Cryptosystems consist of two pieces: the algorithm, or method, for encryption, and a secret piece of information, called the *key*. In the nineteenth century, Auguste Kerckhoffs observed that any cryptosystem used by more than a very small group of people will eventually leak the encryption technique. Thus the secrecy of a system must reside in the key. I assume that the unencrypted message—the *plaintext*—is a string of bits that is to be transformed into an encrypted string: the *ciphertext*. This is done through *Boolean functions*, maps from $\{0,1\}^n$ to $\{0,1\}$ or, more generally, to $\{0,1\}^m$.

In 1949 Claude Shannon considered the question of designing a code that could not be broken [21]. Using information-theoretic techniques he proved that aside from one-time pads—systems that use XOR (bitwise addition mod 2) or other invertible functions that combine the plaintext with a key of the same length—there are no unbreakable cryptosystems.

I must make a minor, but important, digression. In this review of cryptography books, I am concentrating on the mathematics of cryptography. Yet typically, when a cryptosystem is broken, the break occurs as a result of a "side" issue — one-time pads used more than once, keys unsafely stored, insider attacks, etc. So when one speaks of the "unbreakability" of one-time pads, one is speaking only to the mathematical unbreakability of the cryptography and not to the underlying security system in which it is embedded.

Shannon observed that it was better to ask whether it was *feasible* to break the code. A cryptosystem is considered good if the time to break the system (e.g., decode a particular message or, better yet, determine the key used for encryption) is reasonably proportional to a brute-force search of the key space.

In the 1970s computer scientists were exploring notions of time, space, and complexity. The notions of $\mathcal{P}$, the set of problems solvable in polynomial time, and $\mathcal{NP}$, the set of problems solvable in non-deterministic polynomial time (essentially those problems that have a solution checkable in polynomial time), were developed. Shannon's notion of feasible computation fit well into this framework. Diffie and Hellman introduced public-key cryptography: enciphering and deciphering computations with keys $\mathcal{E}$ and $\mathcal{D}$ such that computing $\mathcal{D}$ from $\mathcal{E}$ was *computationally infeasible*. In one of those felicitous coincidences that periodically occur in science, Ralph Merkle independently discovered many of these ideas at about the same time [17]; later, he worked with Diffie and Hellman.

Diffie and Hellman proposed an algorithm that has since become known as the Diffie-Hellman key exchange, a method for establishing a private key over an insecure channel. In Diffie-Hellman key exchange, a large prime p and a generator g for the multiplicative group Z/pZ are both known to Alice and Bob. Alice chooses a random number a and computes $g^a \pmod p$, which she sends to Bob. Bob similarly picks a random number b, computes $g^b \pmod p$ and communicates the result of that computation to Alice. Both Alice and Bob are now able to compute $g^{ab} \equiv g^{ba} \pmod p$. Because of the computational difficulty of computing discrete logs (and the reducibility of the Diffie-Hellman problem to computing discrete logs), an eavesdropper who saw both g^a and g^b would nonetheless find it computationally infeasible to determine g^{ab}.

The third, and very important, contribution of the Diffie-Hellman paper was to define *digital signatures*, or *one-way authentication*; this was a concept that had not been anticipated by the GCHQ researchers. Digital signatures are "digital, unforgeable, message-dependent signatures" [7]. Public-key cryptography provides

a technique for digital signatures: if Alice wants to sign a message $\mathcal{M}$ she is sending to Bob, she sends Bob $\mathcal{M}$ "decrypted" under Alice's private key, $\mathcal{D}_{\mathcal{A}}$. Since Bob knows Alice's public key, he can "encrypt" the enciphered message and check that it really came from Alice (since she is presumably the only one who knows her private key).

Digital signatures enable a server to "prove" to a client that the server is the server it claims to be, e.g., the server to which the PC has connected attests that the server belongs to Amazon (and not Fly-Away-Books.com instead). The PC checks the claim through a chain of "digital certificates", each of which attests to the authenticity of the previous element in the chain through digital signatures and public key/private key pairs. The public key of root certificates (a list typically including AOL Time Warner, Entrust, RSA Data Security, VeriSign, etc.) are stored on the user's PC, enabling these proofs of authenticity. Digital signatures are thus the second critical component of Internet commerce.

At the time that Diffie-Hellman key exchange and RSA encryption were proposed, the fastest solutions for the underlying mathematical problems—computing discrete logs (logarithms base a prime) and factoring integers—required exponential time. (The time to solve a problem is measured in terms of the size of the input. Thus the problem size for Diffie-Hellman key exchange is $O(\log p)$, and for RSA encryption it is $O(\log n)$.) But various mathematicians and computer scientists rediscovered algorithms of Maurice Kraitchik's and Allan Cunningham's from the 1920s that provided a faster, probabilistic, solution to both problems [15, p. 60]: the *index calculus* method.

The *index-calculus* method works for groups that possess a certain structure. Suppose G is a group of order n with elements $g_1, \ldots, g_n$ and generator g; given g^a we seek to efficiently determine "a". The index-calculus method proceeds as follows:

(1) Step 1: Collect m identities of the form:

$$\prod_{j=1}^{m} g_j^{a_{ij}} = g^{bj};$$

rewrite the identities as a set of linear congruences:

$$\sum_{j=1}^{m} a_{ij} \ \mathrm{ind}_g g_j \equiv b_i \pmod{n}.$$

(2) Step 2: Solve for $\mathrm{ind}_g g_j$.

Note that these two steps can be pre-computed.

(3) Step 3: To find a, construct:

$$\prod_{j=1}^{m} g_j^{e_j} = ag^e,$$

which gives $\mathrm{ind}_g a = \Sigma_{j=1}^{m} e_j \mathrm{ind}_g g_j - e$.

The index-calculus method works in groups where it is known how to efficiently generate the relations in Step 1 with the b_is "small", that is, less than some bound dependent on n. This includes some finite fields and class groups of imaginary quadratic number-fields [15, p. 61]. The index-calculus method is used in both the quadratic and number-field sieve algorithms for integer factorization, which take

respectively $\exp\sqrt{\log n \log\log n}$ and $\exp((c\log n)^{1/3}(\log\log n)^{2/3})$ (where c depends on which flavor of the number field sieve is implemented). (See [20] for a lovely elementary exposition of the two sieves.)

Elliptic curves are one type of group for which the index-calculus method does not work. This, and the fact that for each prime power q there are many elliptic curve groups E/F_q (as opposed to just a single finite field), inspired Neal Koblitz and Victor Miller, who, in 1985, independently invented Elliptic Curve Cryptography (ECC). Elliptic curves are solutions to equations of the form

$$y^2 + a_1xy + a_3 = x^3 + a_2x^2 + a_4x + a_6$$

and had been extensively studied. Using point addition, one can create a group structure on the points of the curve. In 1984 Rene Schoof had discovered a polynomial-time algorithm for computing the size of the elliptic curve group over any finite field, thus enabling the use of elliptic curves for several problems, including integer factorization, primality testing, and cryptography. In ECC, the easy-to-compute function is point multiplication: given a point P on an elliptic curve E defined over a finite field F_q, compute kP, that is, P added to itself k times. The infeasible—or at least, infeasible by current methods—computation is determining k given kP.

There is an ECC analogue of Diffie-Hellman key exchange in which Alice and Bob send each other the points aP and bP respectively and the key is abP. There are also various Elliptic Curve Digital Signature Algorithms. ECC rapidly became quite important. As Koblitz put it, "At first, elliptic curve cryptography seemed like the sort of notion that would be of practical utility only in the distant future, if at all. However, as often happens in cryptography, the distant future came quickly" [13, p. 131]. The fact that ECC with a smaller key size offers the same security as RSA and Diffie-Hellman makes ECC attractive to manufacturers of devices with small memory and low power, e.g., cell phones. With the National Institute of Standards and Technology publishing a list of recommended elliptic curves for federal government use (http://csrc.nist.gov/publications/fips/fips186-2/fips186-2-change1.pdf), the distant future is here.

Cryptography is at the intersection of engineering and mathematics; one of the important directions of modern cryptographic research has been in building the mathematical foundations. Shannon's work is, of course, seminal. In recent decades complexity theorists have worked to develop a rigorous treatment of the underlying assumptions. As one begins to examine "obvious" concepts such as security or randomness, it quickly becomes apparent that these are neither obvious nor necessarily well-defined.

What does it mean to say a cryptosystem is secure (recall that I am confining the issue to the mathematics of the cryptography)? If, given the ciphertext, one can guess a single bit of the plaintext with better than 1/2 probability, is the system still secure (or does that single break enable discovering other bits of the plaintext with better than 1/2 probability)? If the system reveals any information about the plaintext, is it still secure (cf. the issue of semantic security discussed in RSA above)? In terms of computational complexity, what does it mean to classify one problem as "simpler" than another? What does randomness mean? Can we, in polynomial time, distinguish between random and pseudorandom sequences? Is there a way to expand short, randomly selected "seeds" into longer, pseudorandom bit sequences?

During the 1980s and 1990s researchers developed different ways of distinguishing security. Zero-knowledge proofs, proofs that reveal that the prover "knows" X without actually revealing X (for example, that a graph can be three colored—no two adjacent vertices with the same color—without revealing anything about any particular three coloring), came from this work. So did $\mathcal{PCP}$, probabilistically checkable proofs, which are proofs checkable in probabilistic polynomial time. $\mathcal{PCP}$ gave a new characterization of the important class $\mathcal{NP}$, which typically is described as the set of membership problems with polynomial-size proofs. (Composite integers and graphs that are three colorable are in $\mathcal{NP}$.) The alternate $\mathcal{PCP}$ characterization of $\mathcal{NP}$ says that finding approximate solutions (say within a constant factor of optimal) for certain $\mathcal{NP}$-optimization problems is $\mathcal{NP}$-hard. As a result of this characterization, there has been great progress in developing polynomial-time algorithms to approximate problems as closely as possible (assuming that $\mathcal{P} \neq \mathcal{NP}$).

Public key is mathematically delightful—and it provides an excellent refutation to Hardy's claim about the inapplicability of pure mathematics—and essential for key exchange, but public-key cryptography has its limitations. In particular, current public-key methods are inefficient and as a result are typically used only for key exchange and signatures. Symmetric-, or private-key, cryptography is usually used for message encryption.

In the mid 1970s, the U.S. government issued a call for a public symmetric-key cryptosystem that would be used for securing sensitive unclassified information [23]. IBM's design was accepted, and the algorithm, which became known as the Data Encryption Standard (DES), was widely adopted, becoming, for example, the cryptographic standard for Electronic Funds Transfer.

DES's acceptance was not without controversy, not the least because of its short key length: 56 bits. Diffie and Hellman argued the algorithm was easily subject to a "brute-force" attack in which an opponent simply tried all possible keys until determining the correct one [8]. In fact, DES did not fall to a brute-force attack until the summer of 1998, when a special-purpose $250,000 computer built by the Electronic Frontier Foundation decrypted a DES-encoded message in 56 hours. Five years earlier, however, two mathematically interesting attacks on DES were developed: differential and linear cryptanalysis. Differential cryptanalysis, discovered by Adi Shamir and Eli Biham, uses differences in the plaintext and how they are handled by the cipher to determine information about the key bits [1]. Linear cryptanalysis, discovered by Mitsuru Matsui, works by finding linear relations between the input and output bits and chaining such relations together [14]. Although in theory differential and linear cryptanalysis represented serious attacks on DES, in practice they did not. The 2^{47} and 2^{43} DES encryptions needed by differential and linear cryptanalysis respectively take sufficiently much time that the EFF brute-force search of the key space was much faster.

Although symmetric-key encryption, the more obvious way of sending secret messages, has always been the mainstay of cryptography, it is only in the last few decades that the field has really benefited from a serious mathematical analysis. The invention of public-key cryptography and the approval of DES as a Federal Information Processing Standard by the U.S. government generated much excitement and the field has blossomed. In 1981 fewer than fifty researchers attended the first open meeting on cryptography, which occurred in Santa Barbara. Now there

are several international meetings a year, numerous workshops, and active research groups in Europe and Asia as well as across North America.

By the late 1990s it was clear that DES would need to be replaced. In January 1997 NIST announced a competition for a symmetric-key algorithm running on 128-bit blocks of data using 128-, 192-, or 256-bit keys (DES has 64-bit block size). Cryptographers from around the world submitted candidates. After several international meetings and two-and-a-half years of of evaluation, NIST announced its choice for the Advanced Encryption Standard: Rijndael, an algorithm written by two Belgian cryptographers, Joan Daemen and Vincent Rijmen [5].

Rijndael is very interesting. It is written as a mathematician might write an algorithm, its steps described algebraically. At some level, this is no surprise. After all, although Rijndael is a transformation from $\{0,1\}^{128}$ to itself, this function can also be described as a map from $GF(2^7)$ to itself, which clearly can be written as an algebraic function. But the special part of Rijndael is the simplicity of the algebraic description. These same functions help provide "proofs" of the algorithm's security ("proofs" is in quotes because these demonstrate security only against known forms of attack, not undiscovered ones).

Rijndael was approved as the Advanced Encryption Standard (AES) in 2001. In June 2003, the U.S. government stamped a higher seal of approval on the algorithm. AES was approved for use at the 128-bit key level in transmitting "SECRET" documents (and at the 192-bit level, "TOP SECRET") [19]. Times have changed. The U.S. government has approved for use in "TOP SECRET" communications an algorithm developed by a community that did not exist thirty years ago. That is indicative of major progress since the early seventies.

Of late there has been a real boom in cryptography books. For example, Springer-Verlag has published six texts in the last several years (this count does not include the proceedings of various conferences). In writing this review, I have tried to confine myself to books most likely to be of interest to mathematicians interested in learning about cryptography. I have omitted elementary books as well as practical ones, and I have not included books whose main focus is security rather than cryptography.

Many mathematicians have been intrigued by the application of computational number theory, and a number of the texts, including some of the more basic ones, have that emphasis. Richard Mollin has written *RSA and Public-Key Cryptography* [18], which he says is intended for senior math majors. There are some useful aspects to this book. Mollin discusses randomness, some attacks on RSA, and key management, all important issues in cryptography, not all of which are covered in other texts. Unfortunately the book is superficial in many ways and wrong in others. For example, Mollin's reason for why one-way functions have not been shown to exist ("...there is no rigorous definition of the terms 'computationally easy' or 'computationally infeasible'" [18, p. 54]) is false, as is his reasoning about cookies [18, p. 184]. The problems are pedestrian. I would not want to teach—or learn—out of this book.

Hans Delfs and Helmut Knebl have also written a textbook on cryptography, *Introduction to Cryptography,* at the senior major level; their audience is students in computer science, mathematics, and engineering [6]. Like Mollin's book, this text emphasizes public-key cryptography, though symmetric-key systems do appear in Chapter 2. (There is, however, no description of AES; I view this as a serious omission.) There is no cryptanalysis presented in this book, but there is a fair

amount of material on protocols, including commitment schemes, voting schemes, and anonymous electronic cash. However, one drawback for mathematicians is the bareness of explanation; e.g., the description of the Kerberos protocol mentions that the reason for time stamps is to prevent replay attacks, but this point is not explained. A mathematician might do better to learn the protocol material from a book that focuses on such issues, e.g., *Network Security* by Kaufman et al. [12].

My favorite of the current crop of undergraduate books is the second edition of *Cryptography: Theory and Practice* by Douglas Stinson. Of the three books at this level covered in this review, Stinson's is the most complete, covering for example, Shannon's paper (as do Delfs and Knebl), whitening (a useful technique that consists simply of XORing with a subkey at the beginning and end of an algorithm), attacks on RSA, linear and differential cryptanalysis, and a description of AES. My main unhappiness with Stinson's book is his descriptions often ignore the why of certain procedures. Thus Stinson says DES begins and ends with the permutations IP and IP^{-1}, but he doesn't tell us that the reason is to spread the data efficiently. Stinson describes the functions SHIFTROW and MIXCOLUMN in AES, but he doesn't explain that those steps are for diffusion. Cryptography is full of complex little ideas that make the difference between an algorithm that works and one that falls flat on its face. I would prefer a book that gave insight into these when such insight is easily accessible. Stinson skimps.

Stinson's book is dense with material and is definitely more than a one-semester course for undergraduates. Some of the material, including the abstract n-tuples used to describe the cryptosystems, may be a bit much for students. But if I were learning/teaching cryptography for the first time to a class of undergraduate math majors, this is the book I would use.

On the other hand, if I were teaching elliptic curve cryptography, I would have a rather different set of choices. A slightly older book (1998), *Algebraic Aspects of Cryptography* by Neal Koblitz, co-inventor of ECC, is charming and more mathematical than many treatments [13]. Koblitz presents public-key cryptography, complexity, and ECC. This material is covered lightly; Koblitz makes no attempt to have a thorough, let-us-cover-every-scheme-there-is text. Koblitz's presentation is a bit idiosyncratic, but definitely interesting. He presents some relevant complexity theory, including discussions of $\mathcal{BPP}$, bounded probabilistic polynomial time, those languages recognizable (with high probability) by a probabilistic polynomial-time algorithm. He also discusses Brassard's Theorem, which gives complexity-theoretic evidence that an NP-hard cryptosystem is unlikely (Koblitz explains what the theorem's implications to cryptography really are). To those who know Koblitz's work, there will be no surprise that the problems are also quite good.

In *Elliptic Curves: Number Theory and Cryptography,* Larry Washington has written a nice, relatively complete, elementary account of elliptic curves. Their application to cryptography is also there, but ECC is less of a focus of the text than in Koblitz's book. Washington covers Rene Schoof's point-counting algorithm (Koblitz does not). This text is definitely a mathematician's viewpoint on ECC; there is no bit counting, no concern with actual implementations, etc.

For the other type of approach, one should read *Elliptic Curves in Cryptography* by Ian Blake, Gadiel Seroussi, and Nigel Smart [3]. This LMS Lecture Note Series book "summarizes the latest knowledge from both theoretical and practical knowledge of ECC." This lovely book has a thorough coverage of bit-counting issues, something that matters greatly when you are thinking of implementing ECC.

The text covers valuable background research, including Hendrik Lenstra's factoring method and Schoof's point counting algorithm; it also presents various attacks (e.g., the MOV attack). The book is short and doesn't have proofs for all the theorems presented, but the text is clearly written and brings the reader up to date on current research. It is a gem.

The complexity-theoretic end of cryptography has not seen the same number of books as the algorithmic end, although Oded Goldreich has written two books, *Modern Cryptography, Probabilistic Proofs, and Pseudorandomness* and *Foundations of Cryptography: Basic Tools* that summarize much of this area. Of the two, *Foundations* is more centered on cryptography and probably of greater interest to mathematicians interested in learning about cryptography. The text presents complexity research which gives the mathematical underpinnings for cryptography; this includes one-way functions, pseudorandom generators, and zero-knowledge proofs. The material is quite abstract; very little relation to real cryptosystems is presented. A mathematician who did not know the underlying cryptography might have trouble understanding how this material applies to cryptography; Goldreich does not make the connections clear. On the other hand, if a reader wants to learn about foundational work, Goldreich's books are the place to go.

Symmetric-key systems are the workhorses of cryptography, so it is perhaps surprising that there are so few books that focus on symmetric-key systems. It is probably because until recently the area was more engineering than mathematics. Daemen and Rijmen's *The Design of Rijndael* is the only recent text in this important subarea of cryptography. The book is both a pleasure and a disappointment.

Rijndael incorporates new design paradigms: a simple algebraic function in the "S-box" and a diffusion strategy, called wide trails, that combines efficiency with resistance to differential and linear cryptanalysis. *The Design of Rijndael* begins with appropriate background in algebra and combinatorics and then proceeds to a technical specification of the algorithm. This is all quite clearly written. The book continues with a discussion of implementation issues; this is followed by a discussion of design philosophy. Then the book becomes significantly more abstract, moving to a discussion of correlation, difference propagation, the wide-trail strategy, and cryptanalysis.

Because *The Design of Rijndael* coherently pulls together the series of papers that preceded the development of Rijndael, the book will prove quite useful for cryptography researchers. Others may find it heavy going. The notation can be cumbersome, and sometimes the motivation is less than clear. This is too bad. Rijndael is a wonderful algorithm—simple to state and implement, with useful, provable security properties. Daemen and Rijmen have thought clearly and cogently about algorithm design. I would have preferred that their book had been written to appeal to a wider audience. But I am carping. Daemen and Rijmen's book is a useful addition to the canon, and many readers will benefit from it.

No review of cryptography books could be complete without a discussion of the *Handbook*, that is, the *Handbook of Applied Cryptography* by Alfred Menezes, Paul van Oorschot, and Scott Vanstone. Published in 1997—and thus somewhat out of date in this fast-moving field—the *Handbook* is an incredible achievement. Modulo the publication date, the Handbook is complete. If I want to check what problems there were with a proposed system, determine how the variations on a particular algorithm developed, see what research preceded and followed an idea,

I go to the *Handbook*. The *Handbook* has accurate, clear, and correct information. It is wonderful.

The book has drawbacks. The *Handbook* has theorems but no proofs. The *Handbook* is not a textbook, and there are no problems. I would not try to learn cryptography from the *Handbook*. Elliptic curve cryptography is only minimally covered. But, in contrast to other texts, the *Handbook* lists important patents and cryptographic standards. If I were limited to only one cryptography text on my shelves, it would be the *Handbook of Applied Cryptography.*[1]

Cryptography is an active and dynamic field, and in the last few years a number of books have appeared that will make it easier for mathematicians to discover the wonderful results that have appeared in papers, newspapers, and even your SSL-enabled web browser. Read and enjoy.

References

[1] E. Biham and A. Shamir, *Differential Cryptanalysis of the Data Encryption Standard,* Springer-Verlag, New York, 1993. MR **95a:**94007

[2] J. Black and H. Urtubia, "Side-Channel Attacks on Symmetric Encryption Schemes: The Case for Authenticated Encryption", 11th USENIX Security Symposium, 2002, http://www.cs.colorado.edu/~jrblack/papers.html

[3] I. Blake, G. Seroussi, and N. Smart, *Elliptic Curves in Cryptography,* Cambridge University Press, 1999. MR **2001i:**94048

[4] D. Bleichenbacher, B. Kaliski, J. Staddon, "Recent Results on PCKS# 1: RSA Encryption Standard", *RSA Laboratories' Bulletin,* Number 7, June 26, 1998.

[5] J. Daemen and V. Rijmen, *The Design of Rijndael: AES — the Advanced Encryption Standard,* Springer-Verlag, 2002.

[6] H. Delfs and H. Knebl, *Introduction to Cryptography,* Springer-Verlag, 2002. MR **2003m:**94060

[7] W. Diffie and M. Hellman, "New Directions in Cryptography", *IEEE Transactions in Information Theory,* Vol. IT-22, No. 6 (1976). MR **55:**10141

[8] W. Diffie and M. Hellman, "Exhaustive Cryptanalysis of the NBS Data Encryption Standard", *Computer,* Vol. 10, No. 6, June 1977, pp. 74-84.

[9] J. Ellis, *The Possibility of Secure Non-Secret Digital Encryption,* GCHQ-CESG publication, January 1970.

[10] O. Goldreich, *Modern Cryptography, Probabilistic Proofs, and Pseudorandomness,* Springer-Verlag, 1999. MR **2000f:**94029

[11] O. Goldreich, *Foundations of Cryptography: Basic Tools,* Cambridge Univ. Press, 2001. MR **2004a:**94042

[12] C. Kaufman, R. Perlman, and M. Speciner, *Network Security: Private Communication in a Public World,* Prentice Hall, 2002.

[13] N. Koblitz, *Algebraic Aspects of Cryptography,* Springer-Verlag, 1998. MR **2000a:**94012

[14] Mitsuru Matsui, "Linear Cryptanalysis Method for DES Cipher", *Eurocrypt '93*, Springer-Verlag, 1994, pp. 386–397.

[15] K. McCurley, "The Discrete Logarithm Problem", in C. Pomerance, ed., *Cryptology and Computational Number Theory,* American Mathematical Society, Proceedings of Symposia in Applied Mathematics, Vol. 42, 1990, pp. 49-74. MR **92d:**11133

[16] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography,* CRC Press, 1997 (fifth printing, 2001). MR **99g:**94015

[17] R. Merkle, "Secure Communications over Insecure Channels", *Communications of the ACM,* Vol. 21, No. 4 (1978), pp. 294-499.

[18] R. Mollin, *RSA and Public-Key Cryptography,* Chapman and Hall/CRC, 2003.

[19] National Security Agency, Committee on National Security Systems, *National Policy on the Use of the Advanced Encryption Standard (AES) to Protect National Security Systems and National Security Information,* CNSS Policy No. 15, Fact Sheet 1, June 2003.

[1]The book is available free online at http://www.cacr.math.uwaterloo.ca/hac.

[20] C. Pomerance, "A Tale of Two Sieves", *Notices of the American Mathematical Society,* Vol. 43 (1996), pp. 1473-85. MR **97f:**11100
[21] C. Shannon, Communication Theory of Secrecy Systems, *Bell System Technical Journal,* Vol. 28, October 1949, pp. 656-716. MR **11:**258d
[22] D. Stinson, *Cryptography: Theory and Practice,* Chapman and Hall/CRC, 2002. MR **2003c:**94035
[23] United States Department of Commerce, National Bureau of Standards (1977), "Data Encryption Standard", Federal Information Processing Standard Publication No. 46.
[24] S. Vaudenay, "Security Flaws Induced by CBC Padding — Applications to SSL, IPSEC, WTLS, . . . ", *Eurocrypt 2002,* Springer- Verlag, 2002, pp. 534-546.
[25] L. Washington, *Elliptic Curves: Number Theory and Cryptography,* Chapman and Hall/CRC, 2003.

Susan Landau
Sun Microsystems
E-mail address: susan.landau@sun.com