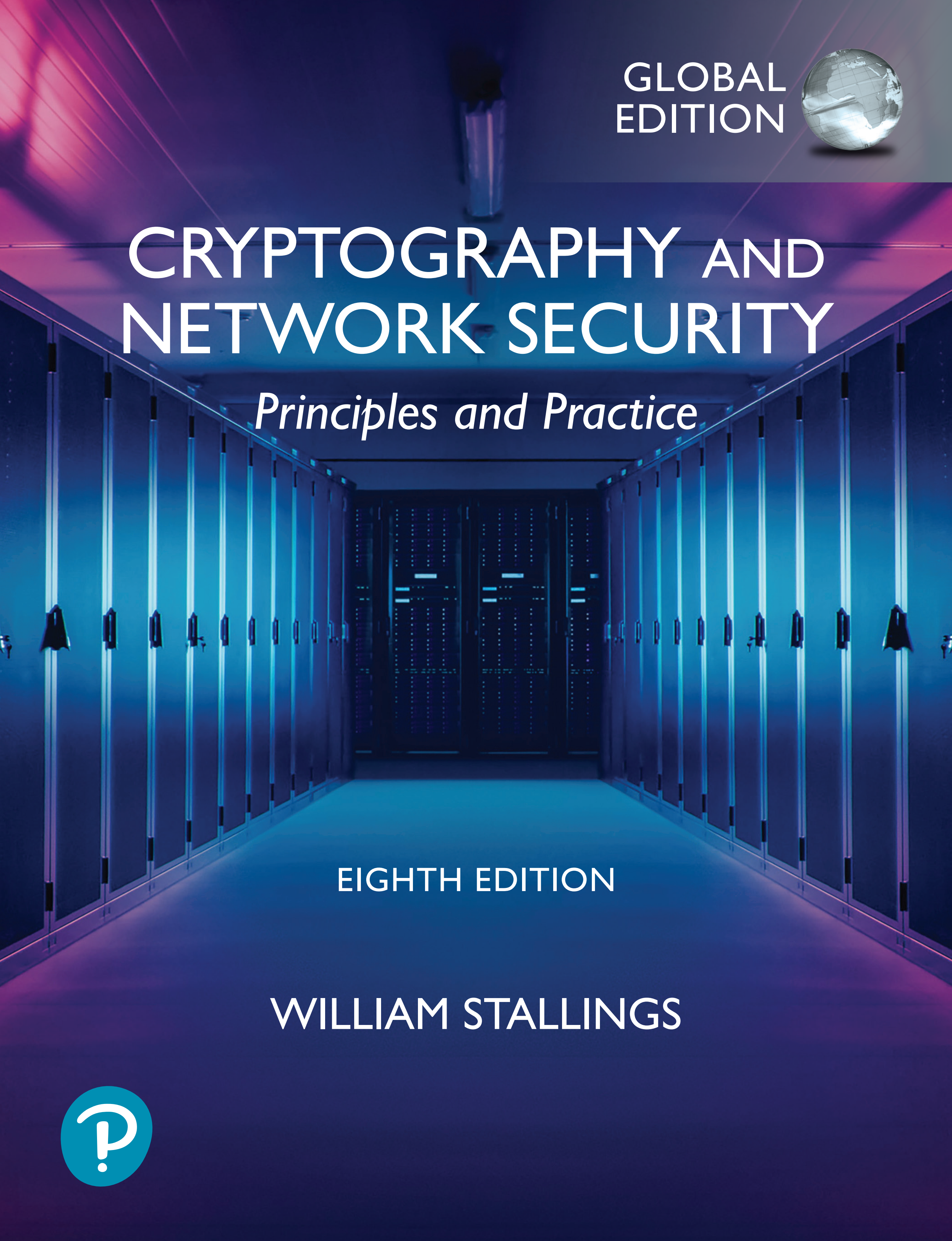
GLOBAL EDITION

CRYPTOGRAPHY AND NETWORK SECURITY

Principles and Practice

EIGHTH EDITION

WILLIAM STALLINGS



Cryptography and Network Security

Principles and Practice

Eighth Edition

Global Edition

William Stallings

Pearson

Product Management: Gargi Banerjee and Paromita Banerjee
Content Strategy: Shabnam Dohutia, Aurko Mitra, Afshaan Khan, and Sharon Thekkekara
Product Marketing: Wendy Gordon, Ashish Jain, and Ellen Harris
Supplements: Bedasree Das
Digital Studio: Vikram Medepalli and Naina Singh
Rights and Permissions: Rimpy Sharma and Akanksha Bhatti
Cover Art: Gorodenkoff / Shutterstock

Credits and acknowledgments borrowed from other sources and reproduced, with permission, in this textbook appear on the appropriate page within text.

Pearson Education Limited
KAO Two
KAO Park
Hockham Way
Harlow
CM17 9SR
United Kingdom

and Associated Companies throughout the world

Visit us on the World Wide Web at: www.pearsonglobaleditions.com

Please contact https://support.pearson.com/getsupport/s/contactsupport with any queries on this content.

ISBN 10: 1-292-43748-0 (print)
ISBN 13: 978-1-292-43748-4 (print)
eBook ISBN 13: 978-1-292-43749-1

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library

1 22

Typeset in Times Ten LT Std by B2R Technologies Pvt. Ltd.

For Tricia: never dull, never boring,
the smartest and bravest person I know

This page is intentionally left blank

Contents

NOTATION

Symbol	Expression	Meaning
D, K	D(K, Y)	Symmetric decryption of ciphertext Y using secret key K
D, PR_a	D(PR_a, Y)	Asymmetric decryption of ciphertext Y using A's private key PR_a
D, PU_a	D(PU_a, Y)	Asymmetric decryption of ciphertext Y using A's public key PU_a
E, K	E(K, X)	Symmetric encryption of plaintext X using secret key K
E, PR_a	E(PR_a, X)	Asymmetric encryption of plaintext X using A's private key PR_a
E, PU_a	E(PU_a, X)	Asymmetric encryption of plaintext X using A's public key PU_a
K		Secret key
PR_a		Private key of user A
PU_a		Public key of user A
MAC, K	MAC(K, X)	Message authentication code of message X using secret key K
GF(p)		The finite field of order p, where p is prime. The field is defined as the set Z_p together with the arithmetic operations modulo p.
GF(2^n)		The finite field of order 2^n
Z_n		Set of nonnegative integers less than n
gcd	gcd(i, j)	Greatest common divisor; the largest positive integer that divides both i and j with no remainder on division.
mod	a mod m	Remainder after division of a by m
mod, $\equiv$	$a \equiv b \pmod{m}$	$a \bmod m = b \bmod m$
mod, $\not\equiv$	$a \not\equiv b \pmod{m}$	$a \bmod m \neq b \bmod m$
dlog	$\text{dlog}_{a,p}(b)$	Discrete logarithm of the number b for the base a (mod p)
φ	$\phi(n)$	The number of positive integers less than n and relatively prime to n. This is Euler's totient function.
Σ	$\sum_{i=1}^{n} a_i$	$a_1 + a_2 + \cdots + a_n$
Π	$\prod_{i=1}^{n} a_i$	$a_1 \times a_2 \times \cdots \times a_n$

Symbol	Expression	Meaning
$\mid$	$i \mid j$	i divides j, which means that there is no remainder when j is divided by i
$\lvert, \rvert$	$\lvert a \rvert$	Absolute value of a
$\Vert$	$x \Vert y$	x concatenated with y
$\approx$	$x \approx y$	x is approximately equal to y
$\oplus$	$x \oplus y$	Exclusive-OR of x and y for single-bit variables; Bitwise exclusive-OR of x and y for multiple-bit variables
$\lfloor, \rfloor$	$\lfloor x \rfloor$	The largest integer less than or equal to x
$\in$	$x \in \mathrm{S}$	The element x is contained in the set S.
$\longleftrightarrow$	$A \longleftrightarrow (a_1, a_2, \ldots a_k)$	The integer A corresponds to the sequence of integers $(a_1, a_2, \ldots a_k)$

Preface

WHAT'S NEW IN THE EIGHTH EDITION

Since the seventh edition of this book was published, the field has seen continued innovations and improvements. In this new edition, I try to capture these changes while maintaining a broad and comprehensive coverage of the entire field. To begin this process of revision, the seventh edition of this book was extensively reviewed by a number of professors who teach the subject and by professionals working in the field. The result is that, in many places, the narrative has been clarified and tightened, and illustrations have been improved.

Beyond these refinements to improve pedagogy and user-friendliness, there have been substantive changes throughout the book. Roughly the same chapter organization has been retained, but much of the material has been revised and new material has been added. The most noteworthy changes are as follows:

- **Trust and trustworthiness:** Chapter 1 includes a new section describing these two concepts, which are key concepts in computer and network security.
- **Stream ciphers:** With the growing importance of stream ciphers, the treatment of stream ciphers has been significantly expanded. There is a new section on stream ciphers based on linear feedback shift registers (LFSRs), and several examples of contemporary stream ciphers are provided.
- **Lightweight cryptography:** The Internet of Things and other small embedded systems require new approaches to cryptography to accommodate the low power requirements, minimum memory, and limited processing power of IoT devices. Two new sections cover this rapidly emerging topic.
- **Post-quantum cryptography:** In anticipation of the potential threat posed by quantum computers, there has been considerable research and development of cryptographic algorithms that are resistant to the threat. Two new sections cover this rapidly emerging topic.

- **Cloud security:** The discussion of cloud security has been expanded, and an entire chapter is devoted to this topic in the new edition.
- **IoT network security:** Similarly, IoT networks have resulted in new requirements for network security protocols, which are covered.

OBJECTIVES

It is the purpose of this book to provide a practical survey of both the principles and practice of cryptography and network security. In the first part of the book, the basic issues to be addressed by a network security capability are explored by providing a tutorial and survey of cryptography and network security technology. The latter part of the book deals with the practice of network security: practical applications that have been implemented and are in use to provide network security.

The subject, and therefore this book, draws on a variety of disciplines. In particular, it is impossible to appreciate the significance of some of the techniques discussed in this book without a basic understanding of number theory and some results from probability theory. Nevertheless, an attempt has been made to make the book self-contained. The book not only presents the basic mathematical results that are needed but provides the reader with an intuitive understanding of those results. Such background material is introduced as needed. This approach helps to motivate the material that is introduced, and the author considers this preferable to simply presenting all of the mathematical material in a lump at the beginning of the book.

SUPPORT OF ACM/IEEE COMPUTER SCIENCE CURRICULA 2013

The book is intended for both academic and professional audiences. As a textbook, it is intended as a one-semester undergraduate course in cryptography and network security for computer science, computer engineering, and electrical engineering majors. This edition supports the recommendations of the ACM/IEEE Computer Science Curricula 2013 (CS2013). CS2013 adds Information Assurance and Security (IAS) to the curriculum recommendation as one of the Knowledge Areas in the Computer Science Body of Knowledge. The document states that IAS is now part of the curriculum recommendation because of the critical role of IAS in computer science education. CS2013 divides all course work into three categories: Core-Tier 1 (all topics should be included in the curriculum), Core-Tier-2 (all or almost all topics should be included), and elective (desirable to provide breadth and depth). In the IAS area, CS2013 recommends topics in Fundamental Concepts and Network Security in Tier 1 and Tier 2, and Cryptography topics as elective. This text covers virtually all of the topics listed by CS2013 in these three categories.

The book also serves as a basic reference volume and is suitable for self-study.

PLAN OF THE TEXT

The book is divided into six parts.

- Background
- Symmetric Ciphers
- Asymmetric Ciphers
- Cryptographic Data Integrity Algorithms
- Mutual Trust
- Network and Internet Security

The book includes a number of pedagogic features, including the use of the computer algebra system Sage and numerous figures and tables to clarify the discussions. Most chapters include a list of key words, review questions, suggestions for further reading, and recommended Web sites. Most chapters also include homework problems. The book also includes an extensive glossary, a list of frequently used acronyms, and a bibliography. In addition, a test bank is available to instructors.

INSTRUCTOR SUPPORT MATERIALS

The major goal of this text is to make it as effective a teaching tool for this exciting and fast-moving subject as possible. This goal is reflected both in the structure of the book and in the supporting material. The text is accompanied by the following supplementary material that will aid the instructor:

- **Solutions manual:** Solutions to all end-of-chapter Review Questions and Problems.
- **Projects manual:** Suggested project assignments for all of the project categories listed below.
- **PowerPoint slides:** A set of slides covering all chapters, suitable for use in lecturing.
- **PDF files:** Reproductions of all figures and tables from the book.
- **Test bank:** A chapter-by-chapter set of questions with a separate file of answers.
- **Supplemental homework problems and solutions:** To aid the student in understanding the material, a separate set of homework problems with solutions are available.

All of these support materials are available at the **Instructor Resource Center (IRC)** for this textbook, which can be reached through the publisher's Web site www.pearsonglobaleditions.com.

PROJECTS AND OTHER STUDENT EXERCISES

For many instructors, an important component of a cryptography or network security course is a project or set of projects by which the student gets hands-on experience to reinforce concepts from the text. This book provides an unparalleled degree of support, including a project's component in the course. The IRC not only includes guidance on how to assign and structure the projects, but also includes a set of project assignments that covers a broad range of topics from the text:

- **Sage projects:** Described in the next section.
- **Hacking project:** Exercise designed to illuminate the key issues in intrusion detection and prevention.
- **Block cipher projects:** A lab that explores the operation of the AES encryption algorithm by tracing its execution, computing one round by hand, and then exploring the various block cipher modes of use. The lab also covers DES. In both cases, an online Java applet is used (or can be downloaded) to execute AES or DES.
- **Lab exercises:** A series of projects that involve programming and experimenting with concepts from the book.
- **Research projects:** A series of research assignments that instruct the student to research a particular topic on the Internet and write a report.
- **Programming projects:** A series of programming projects that cover a broad range of topics and that can be implemented in any suitable language on any platform.
- **Practical security assessments:** A set of exercises to examine current infrastructure and practices of an existing organization.
- **Firewall projects:** A portable network firewall visualization simulator, together with exercises for teaching the fundamentals of firewalls.
- **Case studies:** A set of real-world case studies, including learning objectives, case description, and a series of case discussion questions.
- **Writing assignments:** A set of suggested writing assignments, organized by chapter.
- **Reading/report assignments:** A list of papers in the literature—one for each chapter—that can be assigned for the student to read and then write a short report.
- **Discussion topics:** These topics can be used in a classroom, chat room, or message board environment to explore certain areas in greater depth and to foster student collaboration.

This diverse set of projects and other student exercises enables the instructor to use the book as one component in a rich and varied learning experience and to tailor a course plan to meet the specific needs of the instructor and students.

THE SAGE COMPUTER ALGEBRA SYSTEM

One of the most important features of this book is the use of Sage for cryptographic examples and homework assignments. Sage is an open-source, multiplatform, freeware package that implements a very powerful, flexible, and easily learned mathematics and computer algebra system. Unlike competing systems (such as Mathematica, Maple, and MATLAB), there are no licensing agreements or fees involved. Thus, Sage can be made available on computers and networks at school, and students can individually download the software to their own personal computers for use at home. Another advantage of using Sage is that students learn a powerful, flexible tool that can be used for virtually any mathematical application, not just cryptography.

The use of Sage can make a significant difference to the teaching of the mathematics of cryptographic algorithms. Two documents available at the IRC support student use of Sage. The first document provides a large number of examples of the use of Sage covering many cryptographic concepts. The second document provides exercises in each of these topic areas to enable the student to gain hands-on experience with cryptographic algorithms. This appendix is available to instructors at the IRC for this book. It also includes a section on how to download and get started with Sage, a section on programming with Sage, and exercises that can be assigned to students in the following categories:

- **Chapter 2—Introduction to Number Theory:** Euclidean and extended Euclidean algorithms, polynomial arithmetic, GF(2^4), Euler's Totient function, Miller Rabin, factoring, modular exponentiation, discrete logarithm, and Chinese remainder theorem.
- **Chapter 3—Classical Encryption Techniques:** Affine ciphers and the Hill cipher.
- **Chapter 4—Block Ciphers and the Data Encryption Standard:** Exercises based on SDES.
- **Chapter 6—Advanced Encryption Standard:** Exercises based on SAES.
- **Chapter 8—Random Bit Generation and Stream Ciphers:** Blum Blum Shub, linear congruential generator, and ANSI X9.17 PRNG.
- **Chapter 9—Public-Key Cryptography and RSA:** RSA encrypt/decrypt and signing.
- **Chapter 10—Other Public-Key Cryptosystems:** Diffie-Hellman, elliptic curve.
- **Chapter 11—Cryptographic Hash Functions:** Number-theoretic hash function.
- **Chapter 13—Digital Signatures:** DSA.

ACKNOWLEDGMENTS

This new edition has benefited from review by a number of people who gave generously of their time and expertise. The following people reviewed all or a large part of the manuscript: Hossein Beyzavi (Marymount University), Donald F. Costello (University of Nebraska Lincoln), James Haralambides (Barry University), Tenette Prevatte (Fayetteville Technical Community College), Anand Seetharam (California State University Monterey Bay), Tenette Prevatte (Fayetteville Technical Community College), Marius C. Silaghi (Florida Institute of Technology), Shambhu Upadhyaya (University at Buffalo), Rose Volynskiy (Howard Community College), Katherine Winters (University of Tennessee at Chattanooga), Zhengping Wu (California State University at San Bernardino), Liangliang Xiao (Frostburg State University), Seong-Moo (Sam) Yoo (The University of Alabama in Huntsville), and Hong Zhang (Armstrong State University).

Thanks also to the people who provided detailed technical reviews of one or more chapters: Amaury Behague, Olivier Blazy, Dhananjoy Dey, Matt Frost, Markus Koskinen, Manuel J. Martínez, Veena Nayak, Pritesh Prajapati, Bernard Roussely, Jim Sweeny, Jim Tunnicliffe, and Jose Rivas Vidal.

In addition, I was fortunate to have reviews of individual topics by "subject-area gurus," including Jesse Walker of Intel (Intel's Digital Random Number Generator), Russ Housley of Vigil Security (key wrapping), Joan Daemen (AES), Edward F. Schaefer of Santa Clara University (Simplified AES), Tim Mathews, formerly of RSA Laboratories (S/MIME), Alfred Menezes of the University of Waterloo (elliptic curve cryptography), William Sutton, Editor/Publisher of The Cryptogram (classical encryption), Avi Rubin of Johns Hopkins University (number theory), Michael Markowitz of Information Security Corporation (SHA and DSS), Don Davis of IBM Internet Security Systems (Kerberos), Steve Kent of BBN Technologies (X.509), and Phil Zimmerman (PGP).

Nikhil Bhargava (IIT Delhi) developed the set of online homework problems and solutions. Dan Shumow of Microsoft and the University of Washington developed all of the Sage examples and assignments. Professor Sreekanth Malladi of Dakota State University developed the hacking exercises. Lawrie Brown of the Australian Defence Force Academy provided the AES/DES block cipher projects and the security assessment assignments.

Sanjay Rao and Ruben Torres of Purdue University developed the laboratory exercises that appear in the IRC. The following people contributed project assignments that appear in the instructor's supplement: Henning Schulzrinne (Columbia University); Cetin Kaya Koc (Oregon State University); and David Balenson (Trusted Information Systems and George Washington University). Kim McLaughlin developed the test bank.

Finally, I thank the many people responsible for the publication of this book, all of whom did their usual excellent job. This includes the staff at Pearson, particularly my editor Tracy Johnson and production manager Carole Snyder. Thanks also to the marketing and sales staffs at Pearson, without whose efforts this book would not be in front of you.

ACKNOWLEDGMENTS FOR THE GLOBAL EDITION

Pearson would like to acknowledge and thank the following for their work on the Global Edition.

Contributors

Issteffany Araujo (London Metropolitan University)
George Petrides
Somitra Sanadhya (IIT Jodhpur)
Wen-Nung Tsai (National Yang Ming Chiao Tung University)

Reviewers

Avik Chakraborti (University of Exeter)
Basel Halak (University of Southampton)
Erik Mårtensson (University of Bergen)
Vincent Rijmen (KU Leuven)

About the Author

Dr. William Stallings has authored 18 textbooks, and, counting revised editions, over 70 books on computer security, computer networking, and computer architecture. His writings have appeared in numerous ACM and IEEE publications, including the *Proceedings of the IEEE* and *ACM Computing Reviews*. He has received the award 13 times for the best Computer Science textbook of the year from the Text and Academic Authors Association.

In over 30 years in the field, he has been a technical contributor, technical manager, and an executive with several high-technology firms. He has designed and implemented both TCP/IP-based and OSI-based protocol suites on a variety of computers and operating systems, ranging from microcomputers to mainframes. Currently he is an independent consultant whose clients have included computer and networking manufacturers and customers, software development firms, and leading-edge government research institutions.

He created and maintains the **Computer Science Student Resource Site** at http://www.computersciencestudent.com/. This site provides documents and links on a variety of subjects of general interest to computer science students and professionals. He is a member of the editorial board of *Cryptologia*, a scholarly journal devoted to all aspects of cryptology.

Dr. Stallings holds a PhD from the Massachusetts Institute of Technology in Computer Science and a B.S. from Notre Dame in electrical engineering.

This page is intentionally left blank

PART ONE: BACKGROUND

CHAPTER 1

INFORMATION AND NETWORK SECURITY CONCEPTS

LEARNING OBJECTIVES

After studying this chapter, you should be able to:

- Describe the key security requirements of confidentiality, integrity, and availability.
- Discuss the types of security threats and attacks that must be dealt with and give examples of the types of threats and attacks that apply to different categories of computer and network assets.
- Provide an overview of keyless, single-key, and two-key cryptographic algorithms.
- Provide an overview of the main areas of network security.
- Describe a trust model for information security.
- List and briefly describe key organizations involved in cryptography standards.

This book focuses on two broad areas: cryptography and network security. This overview chapter first looks at some of the fundamental principles of security, encompassing both information security and network security. These include the concepts of security attacks, security services, and security mechanisms. Next, the chapter introduces the two areas of cryptography and network security. Finally, the concepts of trust and trustworthiness are examined.

1.1 CYBERSECURITY, INFORMATION SECURITY, AND NETWORK SECURITY

It would be useful to start this chapter with a definition of the terms cybersecurity, information security, and network security. A reasonably comprehensive definition of cybersecurity is:

Cybersecurity is the protection of information that is stored, transmitted, and processed in a networked system of computers, other digital devices, and network devices and transmission lines, including the Internet. Protection encompasses confidentiality, integrity, availability, authenticity, and accountability. Methods of protection include organizational policies and procedures, as well as technical means such as encryption and secure communications protocols.

As subsets of cybersecurity, we can define the following:

- **Information security:** This term refers to preservation of confidentiality, integrity, and availability of information. In addition, other properties, such as authenticity, accountability, nonrepudiation, and reliability can also be involved.
- **Network security:** This term refers to protection of networks and their service from unauthorized modification, destruction, or disclosure, and provision of assurance that the network performs its critical functions correctly and there are no harmful side effects.

Cybersecurity encompasses information security, with respect to electronic information, and network security. Information security also is concerned with physical (e.g., paper-based) information. However, in practice, the terms cybersecurity and information security are often used interchangeably.

Security Objectives

The cybersecurity definition introduces three key objectives that are at the heart of information and network security:

- **Confidentiality:** This term covers two related concepts:
  - **Data[1] confidentiality:** Assures that private or confidential information is not made available or disclosed to unauthorized individuals.

[1]We can define information as communication or representation of knowledge such as facts, data, or opinions in any medium or form, including textual, numerical, graphic, cartographic, narrative, or audiovisual; and data as information with a specific representation that can be produced, processed, or stored by a computer. Security literature typically does not make much of a distinction, nor does this book.

  - **Privacy:** Assures that individuals control or influence what information related to them may be collected and stored and by whom and to whom that information may be disclosed.

- **Integrity:** This term covers two related concepts:
  - **Data integrity:** Assures that data (both stored and in transmitted packets) and programs are changed only in a specified and authorized manner. This concept also encompasses **data authenticity**, which means that a digital object is indeed what it claims to be or what it is claimed to be, and nonrepudiation, which is assurance that the sender of information is provided with proof of delivery and the recipient is provided with proof of the sender's identity, so neither can later deny having processed the information.
  - **System integrity:** Assures that a system performs its intended function in an unimpaired manner, free from deliberate or inadvertent unauthorized manipulation of the system.
- **Availability:** Assures that systems work promptly and service is not denied to authorized users.

These three concepts form what is often referred to as the **CIA triad**. The three concepts embody the fundamental security objectives for both data and for information and computing services. For example, the NIST standard FIPS 199 (*Standards for Security Categorization of Federal Information and Information Systems*) lists confidentiality, integrity, and availability as the three security objectives for information and for information systems. FIPS 199 provides a useful characterization of these three objectives in terms of requirements and the definition of a loss of security in each category:

- **Confidentiality:** Preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information. A loss of confidentiality is the unauthorized disclosure of information.
- **Integrity:** Guarding against improper information modification or destruction, including ensuring information nonrepudiation and authenticity. A loss of integrity is the unauthorized modification or destruction of information.
- **Availability:** Ensuring timely and reliable access to and use of information. A loss of availability is the disruption of access to or use of information or an information system.

Although the use of the CIA triad to define security objectives is well established, some in the security field feel that additional concepts are needed to present a complete picture (Figure 1.1). Two of the most commonly mentioned are as follows:

- **Authenticity:** The property of being genuine and being able to be verified and trusted; confidence in the validity of a transmission, a message, or message originator. This means verifying that users are who they say they are and that each input arriving at the system came from a trusted source.

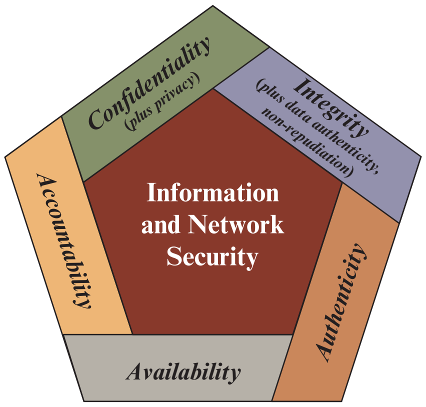


Figure 1.1 Essential Information and Network Security Objectives

- **Accountability:** The security goal that generates the requirement for actions of an entity to be traced uniquely to that entity. This supports nonrepudiation, deterrence, fault isolation, intrusion detection and prevention, and after-action recovery and legal action. Because truly secure systems are not yet an achievable goal, we must be able to trace a security breach to a responsible party. Systems must keep records of their activities to permit later forensic analysis to trace security breaches or to aid in transaction disputes.

The Challenges of Information Security

Information and network security are both fascinating and complex. Some of the reasons follow:

1. Security is not as simple as it might first appear to the novice. The requirements seem to be straightforward; indeed, most of the major requirements for security services can be given self-explanatory, one-word labels: confidentiality, authentication, nonrepudiation, and integrity. But the mechanisms used to meet those requirements can be quite complex, and understanding them may involve rather subtle reasoning.
2. In developing a particular security mechanism or algorithm, one must always consider potential attacks on those security features. In many cases, successful attacks are designed by looking at the problem in a completely different way, therefore exploiting an unexpected weakness in the mechanism.
3. Because of point 2, the procedures used to provide particular services are often counterintuitive. Typically, a security mechanism is complex, and it is not obvious from the statement of a particular requirement that such elaborate measures are needed. It is only when the various aspects of the threat are considered that elaborate security mechanisms make sense.
4. Having designed various security mechanisms, it is necessary to decide where to use them. This is true both in terms of physical placement (e.g., at what points

in a network are certain security mechanisms needed) and in a logical sense [e.g., at what layer or layers of an architecture such as TCP/IP (Transmission Control Protocol/Internet Protocol) should mechanisms be placed].

5. Security mechanisms typically involve more than a particular algorithm or protocol. They also require that participants be in possession of some secret information (e.g., an encryption key), which raises questions about the creation, distribution, and protection of that secret information. There also may be a reliance on communications protocols whose behavior may complicate the task of developing the security mechanism. For example, if the proper functioning of the security mechanism requires setting time limits on the transit time of a message from sender to receiver, then any protocol or network that introduces variable, unpredictable delays may render such time limits meaningless.
6. Information and network security are essentially a battle of wits between a perpetrator who tries to find holes and the designer or administrator who tries to close them. The great advantage that the attacker has is that he or she need only find a single weakness, while the designer must find and eliminate all weaknesses to achieve perfect security.
7. There is a natural tendency on the part of users and system managers to perceive little benefit from security investment until a security failure occurs.
8. Security requires regular, even constant, monitoring, and this is difficult in today's short-term, overloaded environment.
9. Security is still too often an afterthought to be incorporated into a system after the design is complete rather than being an integral part of the design process.
10. Many users and even security administrators view strong security as an impediment to efficient and user-friendly operation of an information system or use of information.

The difficulties just enumerated will be encountered in numerous ways as we examine the various security threats and mechanisms throughout this book.

1.2 THE OSI SECURITY ARCHITECTURE

To assess effectively the security needs of an organization and to evaluate and choose various security products and policies, the manager responsible for security needs some systematic way of defining the requirements for security and characterizing the approaches to satisfying those requirements. This is difficult enough in a centralized data processing environment; with the use of local and wide area networks, the problems are compounded.

ITU-T Recommendation X.800, *Security Architecture for OSI*, defines such a systematic approach. The open systems interconnection (OSI) security architecture is useful to managers as a way of organizing the task of providing security. Furthermore, because this architecture was developed as an international standard, computer and communications vendors have developed security features for their products and services that relate to this structured definition of services and mechanisms.

For our purposes, the OSI security architecture provides a useful, if abstract, overview of many of the concepts that this book deals with. The OSI security architecture focuses on security attacks, mechanisms, and services. These can be defined briefly as:

- **Security attack:** Any action that compromises the security of information owned by an organization.
- **Security mechanism:** A process (or a device incorporating such a process) that is designed to detect, prevent, or recover from a security attack.
- **Security service:** A processing or communication service that enhances the security of the data processing systems and the information transfers of an organization. The services are intended to counter security attacks, and they make use of one or more security mechanisms to provide the service.

In the literature, the terms *threat* and *attack* are commonly used, with the following meanings:

- **Threat:** Any circumstance or event with the potential to adversely impact organizational operations (including mission, functions, image, or reputation), organizational assets, individuals, other organizations, or the Nation through an information system via unauthorized access, destruction, disclosure, modification of information, and/or denial of service.
- **Attack:** Any kind of malicious activity that attempts to collect, disrupt, deny, degrade, or destroy information system resources or the information itself.

The following three sections provide an overview of the concepts of attacks, services, and mechanisms. The key concepts that are covered are summarized in Figure 1.2.

1.3 SECURITY ATTACKS

A useful means of classifying security attacks, used both in X.800, is in terms of *passive attacks* and *active attacks* (Figure 1.2a). A passive attack attempts to learn or make use of information from the system but does not affect system resources. An active attack attempts to alter system resources or affect their operation.

Passive Attacks

Passive attacks are in the nature of **eavesdropping** on, or monitoring of, transmissions. The goal of the attacker is to obtain information that is being transmitted. Two types of passive attacks are the release of message contents and traffic analysis.

The release of message contents is easily understood. A telephone conversation, an electronic mail message, and a transferred file may contain sensitive or confidential information. We would like to prevent an opponent from learning the contents of these transmissions.

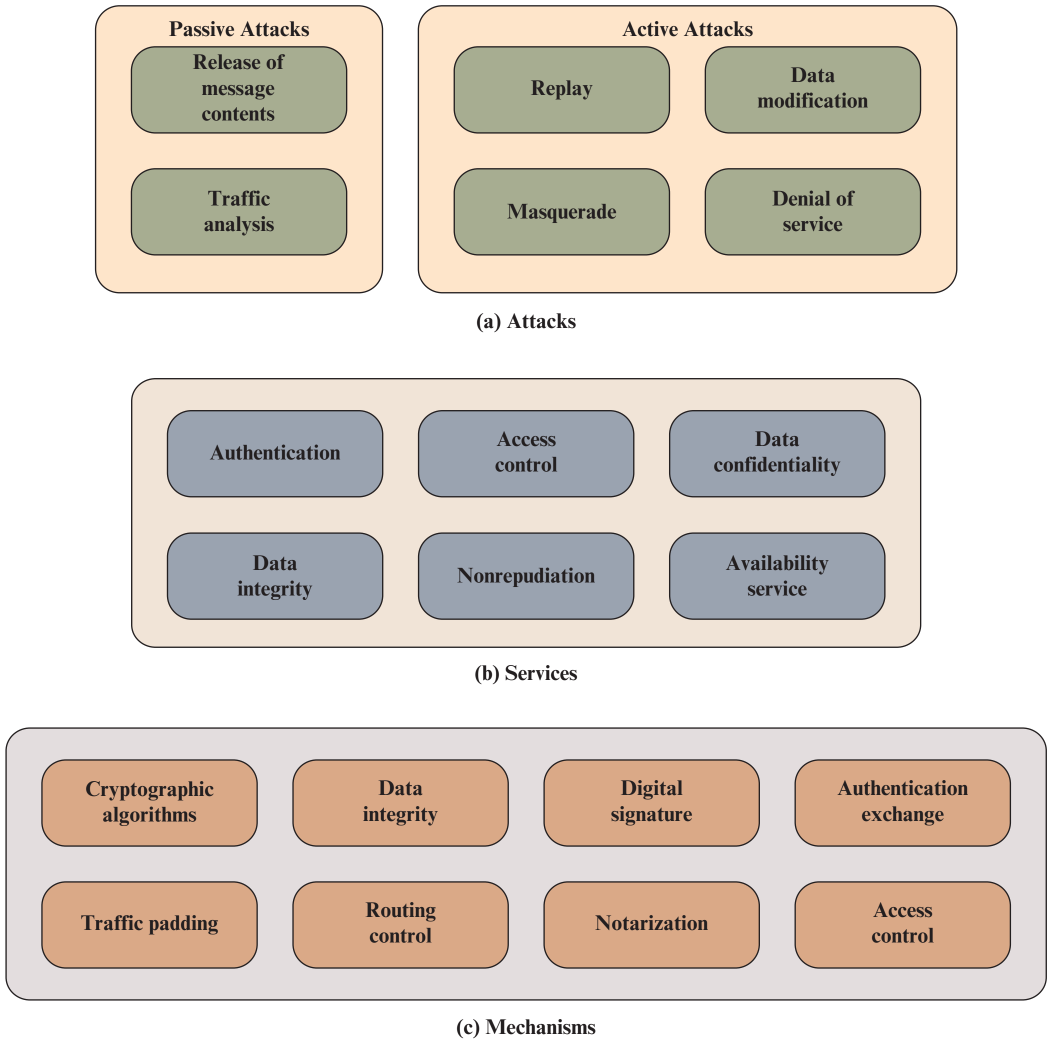


Figure 1.2 Key Concepts in Security

A second type of passive attack, traffic analysis, is subtler. Suppose that we had a way of masking the contents of messages or other information traffic so that opponents, even if they captured the message, could not extract the information from the message. The common technique for masking contents is encryption. If we had encryption protection in place, an opponent might still be able to observe the pattern of these messages. The opponent could determine the location and identity of communicating hosts and could observe the frequency and length of messages being exchanged. This information might be useful in guessing the nature of the communication that was taking place.

Passive attacks are very difficult to detect because they do not involve any alteration of the data. Typically, the message traffic is sent and received in an apparently normal fashion and neither the sender nor receiver is aware that a third party has read the messages or observed the traffic pattern. However, it is feasible to prevent the success of these attacks, usually by means of encryption. Thus, the emphasis in dealing with passive attacks is on prevention rather than detection.

Active Attacks

Active attacks involve some modification of the data stream or the creation of a false stream and can be subdivided into four categories: replay, masquerade, modification of messages, and denial of service.

A **masquerade** takes place when one entity pretends to be a different entity. A masquerade attack usually includes one of the other forms of active attack. For example, authentication sequences can be captured and replayed after a valid authentication sequence has taken place, thus enabling an authorized entity with few privileges to obtain extra privileges by impersonating an entity that has those privileges.

Replay involves the passive capture of a data unit and its subsequent retransmission to produce an unauthorized effect.

Data modification simply means that some portion of a legitimate message is altered, or that messages are delayed or reordered, to produce an unauthorized effect. For example, a message stating, "Allow John Smith to read confidential file accounts" is modified to say, "Allow Fred Brown to read confidential file accounts."

The **denial of service** prevents or inhibits the normal use or management of communication facilities. This attack may have a specific target; for example, an entity may suppress all messages directed to a particular destination (e.g., the security audit service). Another form of service denial is the disruption of an entire network, either by disabling the network or by overloading it with messages so as to degrade performance.

Active attacks present the opposite characteristics of passive attacks. Whereas passive attacks are difficult to detect, measures are available to prevent their success. On the other hand, it is quite difficult to prevent active attacks absolutely, because to do so would require physical protection of all communication facilities and paths at all times. Instead, the goal is to detect them and to recover from any disruption or delays caused by them. Because the detection has a deterrent effect, it may also contribute to prevention.

Figure 1.3 illustrates the types of attacks in the context of a client/server interaction. A passive attack (Figure 1.3b) does not disturb the information flow between the client and server, but is able to observe that flow.

A masquerade can take the form of a man-in-the-middle attack (Figure 1.3c). In this type of attack, the attacker intercepts masquerades as the client to the server and as the server to the client. We see specific applications of this attack in defeating key exchange and distribution protocols (Chapters 10 and 14) and in message authentication protocols (Chapter 11). More generally, it can be used to impersonate the two ends of a legitimate communication. Another form of masquerade is illustrated in Figure 1.3d. Here, an attacker is able to access server resources by masquerading as an authorized user.

Data modification may involve a **man-in-the middle attack**, in which the attacker selectively modifies communicated data between a client and server

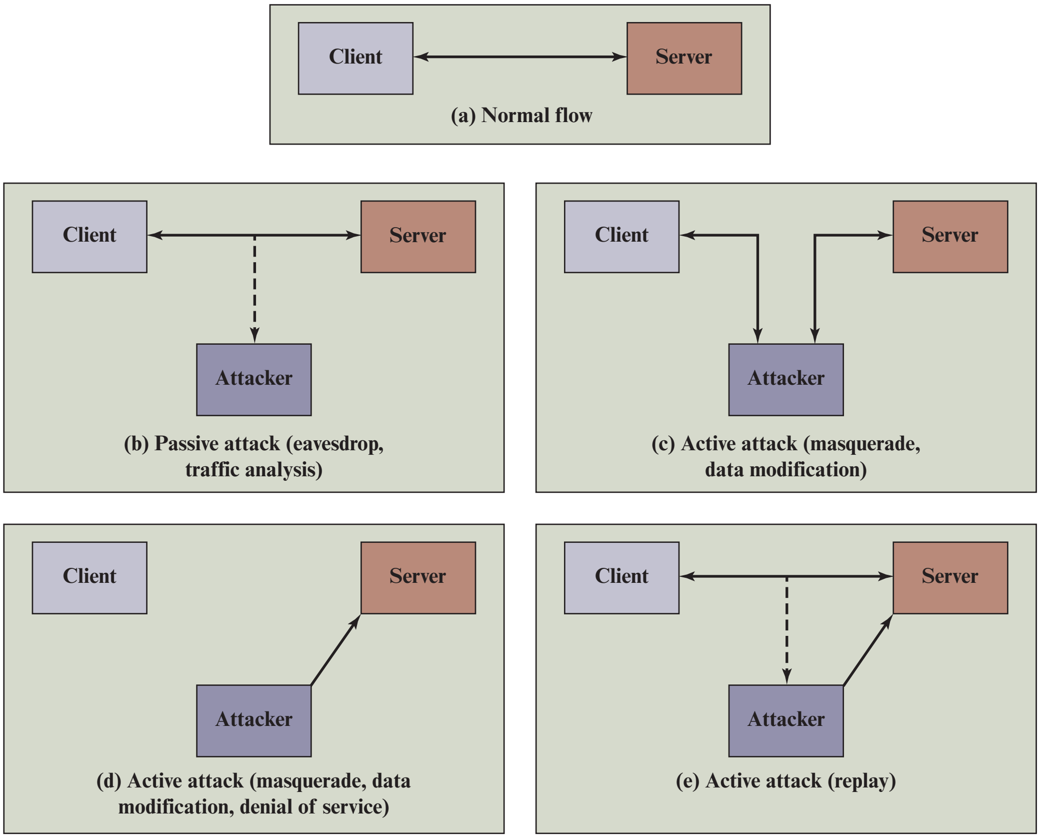

Client
Server
(a) Normal flow
Client
Server
Attacker
(b) Passive attack (eavesdrop, traffic analysis)
Client
Server
Attacker
(c) Active attack (masquerade, data modification)
Client
Server
Attacker
(d) Active attack (masquerade, data modification, denial of service)
Client
Server
Attacker
(e) Active attack (replay)

Figure 1.3 Security Attacks

(Figure 1.3c). Another form of data modification attack is the modification of data residing on a serve or other system after an attacker gains unauthorized access (Figure 1.3d).

Figure 1.3e illustrates the replay attack. As in a passive attack, the attacker does not disturb the information flow between client and server, but does capture client message. The attacker can then subsequently replay any client message to the server.

Figure 1.3d also illustrates denial of service in the context of a client/server environment. The denial of service can take two forms: (1) flooding the server with an overwhelming amount of data; and (2) triggering some action on the server that consumes substantial computing resources.

1.4 SECURITY SERVICES

A security service is a capability that supports one or more of the security requirements (confidentiality, integrity, availability, authenticity, and accountability). Security services implement security policies and are implemented by security mechanisms.

The most important security services are shown in Figure 1.2b. We look at each category in turn.[2]

Authentication

The **authentication** service is concerned with assuring that a communication is authentic. In the case of a single message, such as a warning or alarm signal, the function of the authentication service is to assure the recipient that the message is from the source that it claims to be from. In the case of an ongoing interaction, such as the connection of a client to a server, two aspects are involved. First, at the time of connection initiation, the service assures that the two entities are authentic, that is, that each is the entity that it claims to be. Second, the service must assure that the connection is not interfered with in such a way that a third party can masquerade as one of the two legitimate parties for the purposes of unauthorized transmission or reception.

Two specific authentication services are defined in X.800:

- **Peer entity authentication:** Provides for the corroboration of the identity of a peer entity in an association. Two entities are considered peers if they implement the same protocol in different systems; for example, two TCP modules in two communicating systems. Peer entity authentication is provided for use at the establishment of, or at times during the data transfer phase of, a connection. It attempts to provide confidence that an entity is not performing either a masquerade or an unauthorized replay of a previous connection.
- **Data origin authentication:** Provides for the corroboration of the source of a data unit. It does not provide protection against the duplication or modification of data units. This type of service supports applications like electronic mail, where there are no ongoing interactions between the communicating entities.

Access Control

In the context of network security, access control is the ability to limit and control the access to host systems and applications via communications links. To achieve this, each entity trying to gain access must first be identified, or authenticated, so that access rights can be tailored to the individual.

Data Confidentiality

Confidentiality is the protection of transmitted data from passive attacks. With respect to the content of a data transmission, several levels of protection can be identified. The broadest service protects all user data transmitted between two users

[2]There is no universal agreement about many of the terms used in the security literature. For example, the term *integrity* is sometimes used to refer to all aspects of information security. The term *authentication* is sometimes used to refer both to verification of identity and to the various functions listed under integrity in this chapter. Our usage here agrees with X.800.

over a period of time. For example, when a TCP connection is set up between two systems, this broad protection prevents the release of any user data transmitted over the TCP connection. Narrower forms of this service can also be defined, including the protection of a single message or even specific fields within a message. These refinements are less useful than the broad approach and may even be more complex and expensive to implement.

The other aspect of confidentiality is the protection of traffic flow from analysis. This requires that an attacker not be able to observe the source and destination, frequency, length, or other characteristics of the traffic on a communications facility.

Data Integrity

As with confidentiality, integrity can apply to a stream of messages, a single message, or selected fields within a message. Again, the most useful and straightforward approach is total stream protection.

A connection-oriented integrity service, one that deals with a stream of messages, assures that messages are received as sent with no duplication, insertion, modification, reordering, or replays. The destruction of data is also covered under this service. Thus, the connection-oriented integrity service addresses both message stream modification and denial of service. On the other hand, a connectionless integrity service, one that deals with individual messages without regard to any larger context, generally provides protection against message modification only.

We can make a distinction between service with and without recovery. Because the integrity service relates to active attacks, we are concerned with detection rather than prevention. If a violation of integrity is detected, then the service may simply report this violation, and some other portion of software or human intervention is required to recover from the violation. Alternatively, there are mechanisms available to recover from the loss of integrity of data, as we will review subsequently. The incorporation of automated recovery mechanisms is, in general, the more attractive alternative.

Nonrepudiation

Nonrepudiation prevents either sender or receiver from denying a transmitted message. Thus, when a message is sent, the receiver can prove that the alleged sender in fact sent the message. Similarly, when a message is received, the sender can prove that the alleged receiver in fact received the message.

Availability Service

Availability is the property of a system, or a system resource being accessible and usable upon demand by an authorized system entity, according to performance specifications for the system (i.e., a system is available if it provides services according to the system design whenever users request them). A variety of attacks can result in the loss of or reduction in availability. Some of these attacks are amenable to automated countermeasures, such as authentication and encryption, whereas others require some sort of physical action to prevent or recover from loss of availability of elements of a distributed system.

X.800 treats availability as a property to be associated with various security services. However, it makes sense to call out specifically an availability service. An availability service is one that protects a system to ensure its availability. This service addresses the security concerns raised by denial-of-service attacks. It depends on proper management and control of system resources and thus depends on access control service and other security services.

1.5 SECURITY MECHANISMS

Figure 1.2c lists the most important security mechanisms discussed in this book. These mechanisms will be covered in the appropriate places in the book. So, we do not elaborate now, except to provide the following brief definitions.

- **Cryptographic algorithms:** We can distinguish between reversible cryptographic mechanisms and irreversible cryptographic mechanisms. A reversible cryptographic mechanism is simply an encryption algorithm that allows data to be encrypted and subsequently decrypted. Irreversible cryptographic mechanisms include hash algorithms and message authentication codes, which are used in digital signature and message authentication applications.
- **Data integrity:** This category covers a variety of mechanisms used to assure the integrity of a data unit or stream of data units.
- **Digital signature:** Data appended to, or a cryptographic transformation of, a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery.
- **Authentication exchange:** A mechanism intended to ensure the identity of an entity by means of information exchange.
- **Traffic padding:** The insertion of bits into gaps in a data stream to frustrate traffic analysis attempts.
- **Routing control:** Enables selection of particular physically or logically secure routes for certain data and allows routing changes, especially when a breach of security is suspected.
- **Notarization:** The use of a trusted third party to assure certain properties of a data exchange.
- **Access control:** A variety of mechanisms that enforce access rights to resources.

1.6 CRYPTOGRAPHY

Cryptography is a branch of mathematics that deals with the transformation of data. Cryptographic algorithms are used in many ways in information security and network security. Cryptography is an essential component in the secure storage and transmission of data, and in the secure interaction between parties. Parts Two through Five are devoted to this topic. Here we provide a very brief overview.

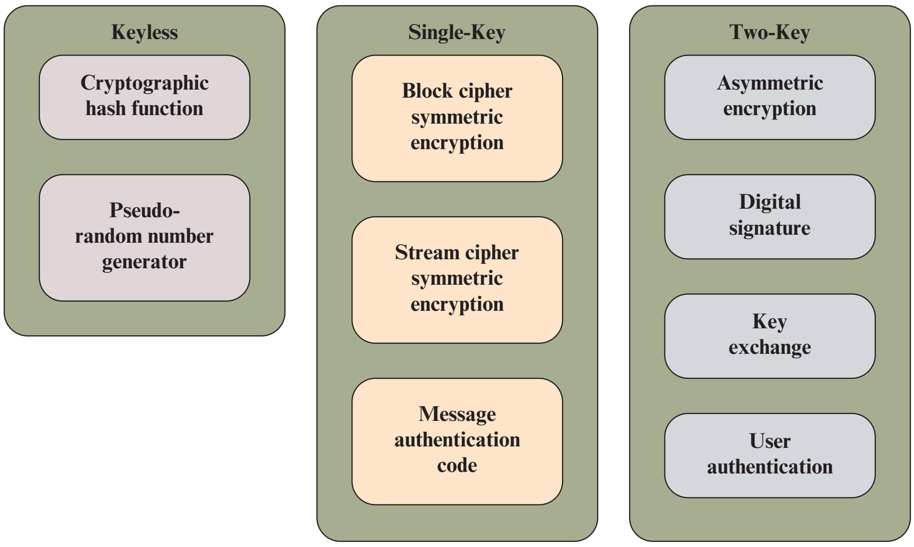

Keyless
Cryptographic hash function
Pseudo-random number generator
Single-Key
Block cipher symmetric encryption
Stream cipher symmetric encryption
Message authentication code
Two-Key
Asymmetric encryption
Digital signature
Key exchange
User authentication

Figure 1.4 Cryptographic Algorithms

Cryptographic algorithms can be divided into three categories (Figure 1.4):

- **Keyless:** Do not use any keys during cryptographic transformations.
- **Single-key:** The result of a transformation is a function of the input data and a single key, known as a secret key.
- **Two-key:** At various stages of the calculation, two different but related keys are used, referred to as a private key and a public key.

Keyless Algorithms

Keyless algorithms are deterministic functions that have certain properties useful for cryptography.

One important type of keyless algorithm is the cryptographic hash function. A hash function turns a variable amount of text into a small, fixed-length value called a *hash value*, *hash code*, or *digest*. A **cryptographic hash function** is one that has additional properties that make it useful as part of another cryptographic algorithm, such as a message authentication code or a digital signature.

A **pseudorandom number generator** produces a deterministic sequence of numbers or bits that has the appearance of being a truly random sequence. Although the sequence appears to lack any definite pattern, it will repeat after a certain sequence length. Nevertheless, for some cryptographic purposes this apparently random sequence is sufficient.

Single-Key Algorithms

Single-key cryptographic algorithms depend on the use of a secret key. This key may be known to a single user; for example, this is the case for protecting stored data that is only going to be accessed by the data creator. Commonly, two parties share the secret key so that communication between the two parties is protected. For certain applications, more than two users may share the same secret key. In this last case, the algorithm protects data from those outside the group who share the key.

Encryption algorithms that use a single key are referred to as **symmetric encryption algorithms**. With symmetric encryption, an encryption algorithm takes as input some data to be protected and a secret key and produces an unintelligible transformation on that data. A corresponding decryption algorithm takes the transformed data and the same secret key and recovers the original data. Symmetric encryption takes the following forms:

- **Block cipher:** A block cipher operates on data as a sequence of blocks. A typical block size is 128 bits. In most versions of the block cipher, known as modes of operation, the transformation depends not only on the current data block and the secret key but also on the content of preceding blocks.
- **Stream cipher:** A stream cipher operates on data as a sequence of bits. Typically, an exclusive-OR operation is used to produce a bit-by-bit transformation. As with the block cipher, the transformation depends on a secret key.

Another form of single-key cryptographic algorithm is the **message authentication code** (MAC). A MAC is a data element associated with a data block or message. The MAC is generated by a cryptographic transformation involving a secret key and, typically, a cryptographic hash function of the message. The MAC is designed so that someone in possession of the secret key can verify the integrity of the message. Thus, the MAC algorithm takes as input a message and secret key and produces the MAC. The recipient of the message plus the MAC can perform the same calculation on the message; if the calculated MAC matches the MAC accompanying the message, this provides assurance that the message has not been altered.

Two-Key Algorithms

Two-key algorithms involve the use of two related keys. A private key is known only to a single user or entity, whereas the corresponding public key is made available to a number of users. Encryption algorithms that use two keys are referred to as **asymmetric encryption algorithms**. Asymmetric encryption can work in two ways:

1. An encryption algorithm takes as input some data to be protected and the private key and produces an unintelligible transformation on that data. A corresponding decryption algorithm takes the transformed data and the corresponding public key and recovers the original data. In this case, only the possessor of the private key can have performed the encryption and any possessor of the public key can perform the decryption.
2. An encryption algorithm takes as input some data to be protected and a public key and produces an unintelligible transformation on that data. A corresponding decryption algorithm takes the transformed data and the corresponding private key and recovers the original data. In this case, any possessor of the public key can have performed the encryption and only the possessor of the private key can perform the decryption.

Asymmetric encryption has a variety of applications. One of the most important is the **digital signature algorithm**. A digital signature is a value computed with a cryptographic algorithm and associated with a data object in such a way that any recipient of the data can use the signature to verify the data's origin and integrity. Typically, the signer of a data object uses the signer's private key to generate the signature, and anyone in possession of the corresponding public key can verify that validity of the signature.

Asymmetric algorithms can also be used in two other important applications. **Key exchange** is the process of securely distributing a symmetric key to two or more parties. **User authentication** is the process of authenticating that a user attempting to access an application or service is genuine and, similarly, that the application or service is genuine. These concepts are explained in detail in subsequent chapters.

1.7 NETWORK SECURITY

Network security is a broad term that encompasses security of the communications pathways of the network and the security of network devices and devices attached to the network (Figure 1.5).

Communications Security

In the context of network security, communications security deals with the protection of communications through the network, including measures to protect against both passive and active attacks (Figure 1.3).

Communications security is primarily implemented using network protocols. A network protocol consists of the format and procedures that governs the transmitting and receiving of data between points in a network. A protocol defines the structure of the individual data units (e.g., packets) and the control commands that manage the data transfer.

With respect to network security, a security protocol may be an enhancement that is part of an existing protocol or a standalone protocol. Examples of the former are IPsec, which is part of the Internet Protocol (IP) and IEEE 802.11i, which is part of the IEEE 802.11 Wi-Fi standard. Examples of the latter are Transport Layer Security (TLS) and Secure Shell (SSH). Part Six examines these and other secure network protocols.

One common characteristic of all of these protocols is that they use a number of cryptographic algorithms as part of the mechanism to provide security.

Device Security

The other aspect of network security is the protection of network devices, such as routers and switches, and end systems connected to the network, such as client systems and servers. The primary security concerns are intruders that gain access to the system to perform unauthorized actions, insert malicious software (malware), or overwhelm system resources to diminish availability. Three types of device security are noteworthy:

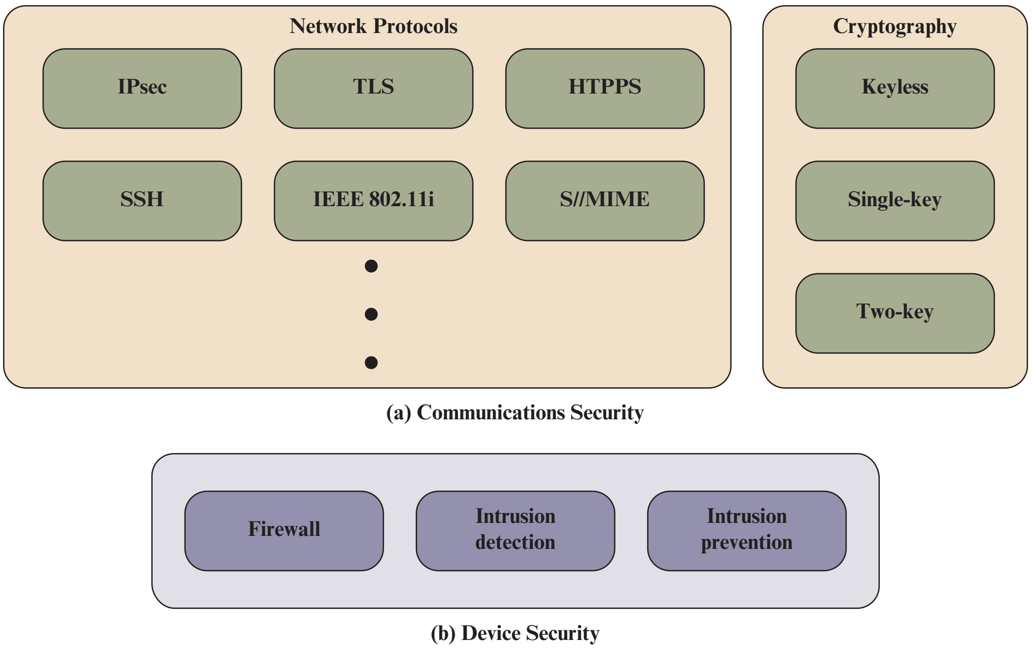


Figure 1.5 Key Elements of Network Security

- **Firewall:** A hardware and/or software capability that limits access between a network and devices attached to the network, in accordance with a specific security policy. The firewall acts as a filter that permits or denies data traffic, both incoming and outgoing, using a set of rules based on traffic content and/ or traffic pattern.
- **Intrusion detection:** Hardware or software products that gather and analyze information from various areas within a computer or a network for the purpose of finding, and providing real-time or near-real-time warning of, attempts to access system resources in an unauthorized manner.
- **Intrusion prevention:** Hardware or software products designed to detect intrusive activity and attempt to stop the activity, ideally before it reaches its target.

These device security capabilities are more closely related to the field of computer security than network security. Accordingly, they are dealt with more briefly than communications security in Part Six. For a more detailed treatment, see [STAL18].

1.8 TRUST AND TRUSTWORTHINESS

The concepts of trust and trustworthiness are key concepts in computer and network security [SCHN91]. It will be useful to look first at a generalized model of trust and trustworthiness, and then apply these concepts to the topic of information security.

A Trust Model

One of the most widely accepted and most cited definitions of trust in the organizational science literature is from [MAYE95], which defines **trust** as follows: the willingness of a party to be vulnerable to the actions of another party based on the expectation that the other will perform a particular action important to the truster, irrespective of the ability to monitor or control that other party.

Three related concepts are relevant to a trust model:

- **Trustworthiness:** A characteristic of an entity that reflects the degree to which that entity is deserving of trust.
- **Propensity to trust:** A tendency to be willing to trust others across a broad spectrum of situations and trust targets. This suggests that every individual has some baseline level of trust that will influence the person's willingness to rely on the words and actions of others.
- **Risk:** A measure of the extent to which an entity is threatened by a potential circumstance or event, and typically a function of 1) the adverse impacts that would arise if the circumstance or event occurs; and 2) the likelihood of occurrence.

Figure 1.6, adapted from [MAYE95], illustrates the relationship among these concepts. Trust is a function of the truster's propensity to trust and the perceived trustworthiness of the trustee. Propensity can also be expressed as the level of risk that an entity (individual or organization) is prepared to tolerate.

Typically, a truster uses a number of factors to establish the trustworthiness of an entity. Three general factors are commonly cited:

- **Ability:** Also referred to as *competence*, this relates to the potential ability of the evaluated entity to do a given task or be entrusted with given information.
- **Benevolence:** This implies a disposition of goodwill towards the trusting party. That is, a trustworthy party does not intend to cause harm to the trusting party.
- **Integrity:** This can be defined as the truster's perception that the trustee adheres to a set of principles that the truster finds acceptable. Integrity implies that a benevolent party takes such measures are necessary to assure that it in fact does not cause harm to the trusting party.

The goal of trust, in the model of Figure 1.6, is to determine what course of action, if any, the trusting party is willing to take in relation to the trusted party. Based on the level of trust, and the perceived risk, the trusting party may decide to take some action that involves some degree of risk taking. The outcome of the risk taking could be a reliance on the trusted party to perform some action or the disclosure of information to the trusted party with the expectation that the information will be protected as agreed between the parties.

The Trust Model and Information Security

Trust is confidence that an entity will perform in a way the will not prejudice the security of the user of the system of which that entity is a part. Trust is always restricted to specific functions or ways of behavior and is meaningful only in the

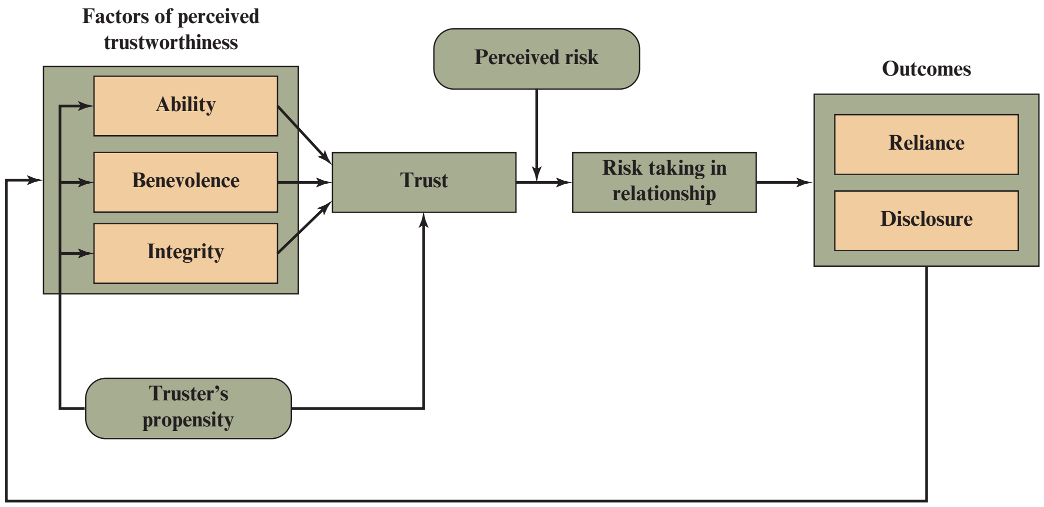


Figure 1.6 Trust Model

context of a security policy. Generally, an entity is said to trust a second entity when the first entity assumes that the second entity will behave exactly as the first entity expects. This trust may apply only for some specific function. In this context, the term *entity* may refer to a single hardware component or software module, a piece of equipment identified by make and model, a site or location, or an organization.

Trustworthiness of an Individual Organizations need to be concerned about both internal users (employees, on-site contractors) and external users (customers, suppliers) of their information systems. With respect to internal users, an organization develops a level of trust in individuals by policies in the following two areas [STAL19]:

- **Human resource security:** Sound security practice dictates that information security requirements be embedded into each stage of the employment life cycle, specifying security-related actions required during the induction of each individual, their ongoing management, and termination of their employment. Human resource security also includes assigning ownership of information (including responsibility for its protection) to capable individuals and obtaining confirmation of their understanding and acceptance.
- **Security awareness and training:** This area refers to disseminating security information to all employees, including IT staff, IT security staff, and management, as well as IT users and other employees. A workforce that has a high level of security awareness and appropriate security training for each individual's role is as important, if not more important, than any other security countermeasure or control.

For external users, trust will depend on the context. In general terms, the factors of perceived trustworthiness and the truster's propensity, as depicted in Figure 1.6, determine the level of trust. Further, the issue of trust is mutual. That is, not only must an organization determine a level of trust towards external users, but external users need to be concerned about the degree to which they can trust an information resource that they use. This mutual trust involves a number a practical consequences, including the use of a public-key infrastructure and user authentication protocols. These matters are explored in Part Five.

Trustworthiness of an Organization Most organizations rely, to a greater or lesser extent, on information system service and information provided by external organizations, as well as partnerships to accomplish missions and business functions. Examples are cloud service providers and companies that form part of the supply chain for the organization. To manage risk to the organization, it must establish trust relationships with these external organizations. NIST SP 800-39 (*Managing Information Security Risk*, March 2011) indicates that such trust relationships can be:

- Formally established, for example, by documenting the trust-related information in contracts, service-level agreements, statements of work, memoranda of agreement/understanding, or interconnection security agreements;
- Scalable and inter-organizational or intra-organizational in nature; and/or
- Represented by simple (bilateral) relationships between two partners or more complex many-to-many relationships among many diverse partners.

The requirements for establishing and maintaining trust depend on mission/business requirements, the participants involved in the trust relationship, the criticality/sensitivity of the information being shared or the types of services being rendered, the history between the organizations, and the overall risk to the organizations participating in the relationship.

As with individuals, trust related to organizations can involve the use of public-key infrastructure and user authentication, as well as the network security measures described in Part Six.

Trustworthiness of Information Systems SP 800-39 defines trustworthiness for information systems as the degree to which information systems (including the information technology products from which the systems are built) can be expected to preserve the confidentiality, integrity, and availability of the information being processed, stored, or transmitted by the systems across the full range of threats. Two factors affecting the trustworthiness of information systems are:

- **Security functionality:** The security features/functions employed within the system. These include cryptographic and network security technologies discussed throughout this book.
- **Security assurance:** The grounds for confidence that the security functionality is effective in its application. This area is addressed by security management techniques, such as auditing and incorporating security considerations into the system development life cycle [STAL19].

Establishing Trust Relationships

The methods used by an organization to establish a **trust relationship** with various entities will depend on a variety of factors, such as laws and regulations, risk tolerance, and the criticality and sensitivity of the relationship. SP 800-39 describes the following methods:

- **Validated trust:** Trust is based on evidence obtained by the trusting organization about the trusted organization or entity. The information may include information security policy, security measures, and level of oversight. An example would be for one organization to develop an application or information system and provide evidence (e.g., security plan, assessment results) to a second organization that supports the claims by the first organization that the application/system meets certain security requirements and/or addresses the appropriate security controls.
- **Direct historical trust:** This type of trust is based on the security-related track record exhibited by an organization in the past, particularly in interactions with the organization seeking to establish trust.
- **Mediated trust:** Mediated trust involves the use of a third party that is mutually trusted by two parties, with the third party providing assurance or guarantee of a given level of trust between the first two parties. An example of this form of trust establishment is the use of public-key certificate authorities, described in Chapter 14.
- **Mandated trust:** An organization establishes a level of trust with another organization based on a specific mandate issued by a third party in a position of authority. For example, an organization may be given the responsibility and the authority to issue public key certificates for a group of organizations.

An organization is likely to use a combination of these methods to establish relationships with a number of other entities.

1.9 STANDARDS

Many of the security techniques and applications described in this book have been specified as standards. Additionally, standards have been developed to cover management practices and the overall architecture of security mechanisms and services. Throughout this book, we describe the most important standards in use or being developed for various aspects of cryptography and network security. Various organizations have been involved in the development or promotion of these standards. The most important (in the current context) of these organizations are as follows:

- **National Institute of Standards and Technology:** NIST is a U.S. federal agency that deals with measurement science, standards, and technology related to U.S. government use and to the promotion of U.S. private-sector innovation. Despite its national scope, NIST Federal Information Processing Standards (FIPS) and Special Publications (SP) have a worldwide impact.
- **Internet Society:** ISOC is a professional membership society with worldwide organizational and individual membership. It provides leadership in addressing issues that confront the future of the Internet and is the organization home for the groups responsible for Internet infrastructure standards, including the Internet Engineering Task Force (IETF) and the Internet Architecture Board (IAB). These organizations develop Internet standards and related specifications, all of which are published as Requests for Comments (RFCs).

- **ITU-T:** The International Telecommunication Union (ITU) is an international organization within the United Nations System in which governments and the private sector coordinate global telecom networks and services. The ITU Telecommunication Standardization Sector (ITU-T) is one of the three sectors of the ITU. ITU-T's mission is the development of technical standards covering all fields of telecommunications. ITU-T standards are referred to as Recommendations.
- **ISO:** The International Organization for Standardization (ISO) is a worldwide federation of national standards bodies from more than 140 countries, one from each country. ISO is a nongovernmental organization that promotes the development of standardization and related activities with a view to facilitating the international exchange of goods and services and to developing cooperation in the spheres of intellectual, scientific, technological, and economic activity. ISO's work results in international agreements that are published as International Standards.

1.10 KEY TERMS, REVIEW QUESTIONS, AND PROBLEMS

Key Terms

access control	digital signature algorithms	pseudorandom number generator
active attack	eavesdropping	replay
asymmetric encryption algorithms	encryption	routing control
attack	firewall	security attack
authentication	information security	security mechanism
authentication exchange	intrusion detection	security service
authenticity	intrusion prevention	single-key algorithm
availability	key exchange	stream cipher
block cipher	keyless algorithm	symmetric encryption algorithms
confidentiality	man-in-the-middle attack	system integrity
cryptographic hash function	masquerade	threat
cryptography	message authentication code	trust
cybersecurity	network security	trust relationship
data authenticity	notarization	trustworthiness
data confidentiality	OSI security architecture	two-key algorithm
data integrity	passive attack	user authentication
data origin authentication	peer entity authentication	
denial of service	privacy	

Review Questions

1.1 What is the OSI security architecture?
1.2 List and briefly define the three key objectives of computer security.
1.3 List and briefly define categories of passive and active security attacks.
1.4 List and briefly define categories of security services.
1.5 List and briefly define categories of security mechanisms.
1.6 List and briefly define the fundamental security design principles.

1.7 Provide an overview of the three types of cryptographic algorithms.
1.8 Provide an overview of the two major elements of network security.
1.9 Briefly explain the concepts of trust and trustworthiness.

Problems

1.1 Consider an automated cash deposit machine in which users provide a card or an account number to deposit cash. Give examples of confidentiality, integrity, and availability requirements associated with the system, and, in each case, indicate the degree of importance of the requirement.

1.2 Repeat Problem 1.1 for a payment gateway system where a user pays for an item using their account via the payment gateway.

1.3 Consider a financial report publishing system used to produce reports for various organizations.
   a. Give an example of a type of publication for which confidentiality of the stored data is the most important requirement.
   b. Give an example of a type of publication in which data integrity is the most important requirement.
   c. Give an example in which system availability is the most important requirement.

1.4 For each of the following assets, assign a low, moderate, or high impact level for the loss of confidentiality, availability, and integrity, respectively. Justify your answers.
   a. A student maintaining a blog to post public information.
   b. An examination section of a university that is managing sensitive information about exam papers.
   c. An information system in a pathological laboratory maintaining the patient's data.
   d. A student information system used for maintaining student data in a university that contains both personal, academic information and routine administrative information (not privacy related). Assess the impact for the two data sets separately and the information system as a whole.
   e. A university library contains a library management system, which controls the distribution of books among the students of various departments. The library management system contains both the student data and the book data. Assess the impact for the two data sets separately and the information system as a whole.

1.5 It is useful to read some of the classic tutorial papers on computer security; these provide a historical perspective from which to appreciate current work and thinking. The following are good examples:

— Browne, P. "Computer Security—A Survey." *ACM SIGMIS Database*, Fall 1972.
— LAMP04 Lampson, B. "Computer Security in the Real World," *Computer*, June 2004.
— Saltzer, J., and Schroeder, M. "The Protection of Information in Computer Systems." *Proceedings of the IEEE*, September 1975.
— Shanker, K. "The Total Computer Security Problem: An Overview." *Computer*, June 1977.
— Summers, R. "An Overview of Computer Security." *IBM Systems Journal*, Vol. 23, No. 4, 1984.
— *Ware, W., ed. Security Controls for Computer Systems. RAND Report 609-1. October 1979.*

Read all of these papers. The papers are available at box.com/Crypto8e. Compose a 500–1000 word paper (or 8–12 slide PowerPoint presentation) that summarizes the key concepts that emerge from these papers, emphasizing concepts that are common to most or all of the papers.

CHAPTER 2

INTRODUCTION TO NUMBER THEORY

LEARNING OBJECTIVES

After studying this chapter, you should be able to:

- ◆ Understand the concept of divisibility and the division algorithm.
- ◆ Understand how to use the Euclidean algorithm to find the greatest common divisor.
- ◆ Present an overview of the concepts of modular arithmetic.
- ◆ Explain the operation of the extended Euclidean algorithm.
- ◆ Discuss key concepts relating to prime numbers.
- ◆ Understand Fermat's theorem.
- ◆ Understand Euler's theorem.
- ◆ Define Euler's totient function.
- ◆ Make a presentation on the topic of testing for primality.
- ◆ Explain the Chinese remainder theorem.
- ◆ Define discrete logarithms.

Number theory is pervasive in cryptographic algorithms. This chapter provides sufficient breadth and depth of coverage of relevant number theory topics for understanding the wide range of applications in cryptography. The reader familiar with these topics can safely skip this chapter.

The first three sections introduce basic concepts from number theory that are needed for understanding finite fields; these include divisibility, the Euclidian algorithm, and modular arithmetic. The reader may study these sections now or wait until ready to tackle Chapter 5 on finite fields.

Sections 2.4 through 2.8 discuss aspects of number theory related to prime numbers and discrete logarithms. These topics are fundamental to the design of asymmetric (public-key) cryptographic algorithms. The reader may study these sections now or wait until ready to read Part Three.

The concepts and techniques of number theory are quite abstract, and it is often difficult to grasp them intuitively without examples. Accordingly, this chapter includes a number of examples, each of which is highlighted in a shaded box.

2.1 DIVISIBILITY AND THE DIVISION ALGORITHM

Divisibility

We say that a nonzero b **divides** a if $a = mb$ for some m, where a, b, and m are integers. That is, b divides a if there is no remainder on division. The notation $b|a$ is commonly used to mean b divides a. Also, if $b|a$, we say that b is a **divisor** of a.

The positive divisors of 24 are 1, 2, 3, 4, 6, 8, 12, and 24.
$13 \mid 182$; $-5 \mid 30$; $17 \mid 289$; $-3 \mid 33$; $17 \mid 0$

Subsequently, we will need some simple properties of divisibility for integers, which are as follows:

- If $a \mid 1$, then $a = \pm 1$.
- If $a \mid b$ and $b \mid a$, then $a = \pm b$.
- Any $b \neq 0$ divides 0.
- If $a \mid b$ and $b \mid c$, then $a \mid c$:

$11 \mid 66$ and $66 \mid 198 \Rightarrow 11 \mid 198$

- If $b \mid g$ and $b \mid h$, then $b \mid (mg + nh)$ for arbitrary integers m and n.

To see this last point, note that

- If $b \mid g$, then g is of the form $g = b \times g_1$ for some integer g_1.
- If $b \mid h$, then h is of the form $h = b \times h_1$ for some integer h_1.

So

$$mg + nh = mbg_1 + nbh_1 = b \times (mg_1 + nh_1)$$

and therefore b divides $mg + nh$.

$b = 7$; $g = 14$; $h = 63$; $m = 3$; $n = 2$
$7 \mid 14$ and $7 \mid 63$.
To show $7 \mid (3 \times 14 + 2 \times 63)$,
we have $(3 \times 14 + 2 \times 63) = 7(3 \times 2 + 2 \times 9)$,
and it is obvious that $7 \mid (7(3 \times 2 + 2 \times 9))$.

The Division Algorithm

Given any positive integer n and any nonnegative integer a, if we divide a by n, we get an integer quotient q and an integer remainder r that obey the following relationship:

$$a = qn + r \qquad 0 \leq r < n;\ q = \lfloor a/n \rfloor \tag{2.1}$$

where $\lfloor x \rfloor$ is the largest integer less than or equal to x. Equation (2.1) is referred to as the division algorithm.[1]

[1]Equation (2.1) expresses a theorem rather than an algorithm, but by tradition, this is referred to as the division algorithm.

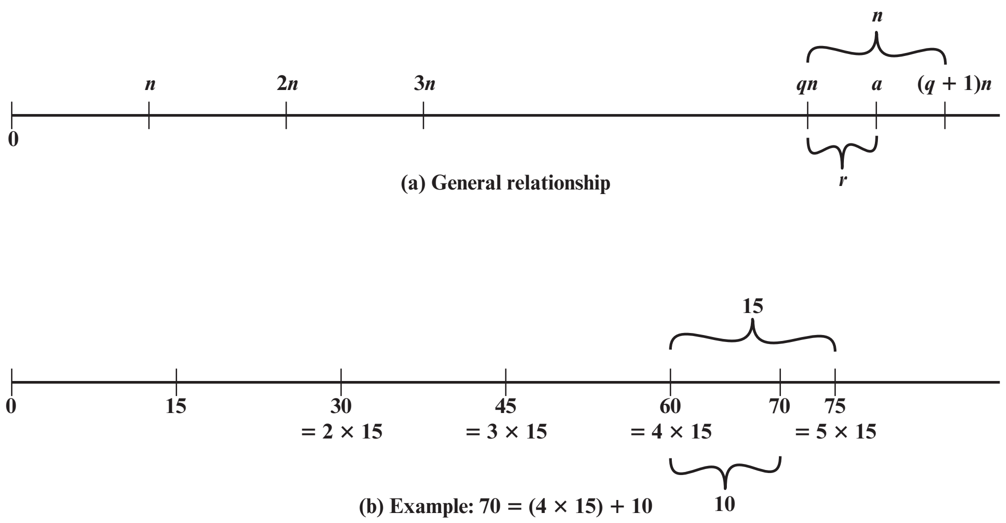


Figure 2.1 The Relationship $a = qn + r; 0 \leq r < n$

Figure 2.1a demonstrates that, given a and positive n, it is always possible to find q and r that satisfy the preceding relationship. Represent the integers on the number line; a will fall somewhere on that line (positive a is shown, a similar demonstration can be made for negative a). Starting at 0, proceed to n, $2n$, up to qn, such that $qn \leq a$ and $(q + 1)n > a$. The distance from qn to a is r, and we have found the unique values of q and r. The remainder r is often referred to as a **residue**.

$a = 11;$	$n = 7;$	$11 = 1 \times 7 + 4;$	$r = 4$	$q = 1$
$a = -11;$	$n = 7;$	$-11 = (-2) \times 7 + 3;$	$r = 3$	$q = -2$

Figure 2.1b provides another example.

2.2 THE EUCLIDEAN ALGORITHM

One of the basic techniques of number theory is the Euclidean algorithm, which is a simple procedure for determining the greatest common divisor of two positive integers. First, we need a simple definition: Two integers are **relatively prime** if and only if their only common positive integer factor is 1.

Greatest Common Divisor

Recall that nonzero b is defined to be a divisor of a if $a = mb$ for some m, where a, b, and m are integers. We will use the notation gcd(a, b) to mean the **greatest common divisor** of a and b. The greatest common divisor of a and b is the largest integer that divides both a and b. We also define gcd(0, 0) $= 0$.

More formally, the positive integer c is said to be the greatest common divisor of a and b if

1. c is a divisor of a and of b.
2. any divisor of a and b is a divisor of c.

An equivalent definition is the following:

$$\gcd(a, b) = \max[k, \text{such that } k|a \text{ and } k|b]$$

Because we require that the greatest common divisor be positive, $\gcd(a, b) = \gcd(a, -b) = \gcd(-a, b) = \gcd(-a, -b)$. In general, $\gcd(a, b) = \gcd(|a|, |b|)$.

$$\boxed{\gcd(60, 24) = \gcd(60, -24) = 12}$$

Also, because all nonzero integers divide 0, we have $\gcd(a, 0) = |a|$.

We stated that two integers a and b are relatively prime if and only if their only common positive integer factor is 1. This is equivalent to saying that a and b are relatively prime if $\gcd(a, b) = 1$.

8 and 15 are relatively prime because the positive divisors of 8 are 1, 2, 4, and 8, and the positive divisors of 15 are 1, 3, 5, and 15. So 1 is the only integer on both lists.

Finding the Greatest Common Divisor

We now describe an algorithm credited to Euclid for easily finding the greatest common divisor of two integers (Figure 2.2). This algorithm has broad significance in cryptography. The explanation of the algorithm can be broken down into the following points:

1. Suppose we wish to determine the greatest common divisor d of the integers a and b; that is determine $d = \gcd(a, b)$. Because $\gcd(|a|, |b|) = \gcd(a, b)$, there is no harm in assuming $a \geq b > 0$.
2. Dividing a by b and applying the division algorithm, we can state:

$$a = q_1 b + r_1 \qquad 0 \leq r_1 < b \tag{2.2}$$

3. First consider the case in which $r_1 = 0$. Therefore b divides a and clearly no larger number divides both b and a, because that number would be larger than b. So we have $d = \gcd(a, b) = b$.
4. The other possibility from Equation (2.2) is $r_1 \neq 0$. For this case, we can state that $d|r_1$. This is due to the basic properties of divisibility: the relations $d|a$ and $d|b$ together imply that $d|(a - q_1 b)$, which is the same as $d|r_1$.
5. Before proceeding with the Euclidian algorithm, we need to answer the question: What is the $\gcd(b, r_1)$? We know that $d|b$ and $d|r_1$. Now take any arbitrary integer c that divides both b and r_1. Therefore, $c|(q_1 b + r_1) = a$. Because c divides both a and b, we must have $c \leq d$, which is the greatest common divisor of a and b. Therefore $d = \gcd(b, r_1)$.

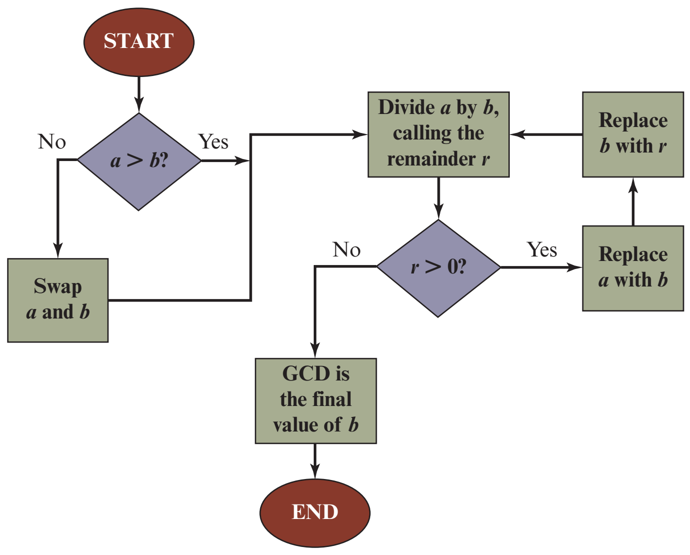


Figure 2.2 Euclidean Algorithm

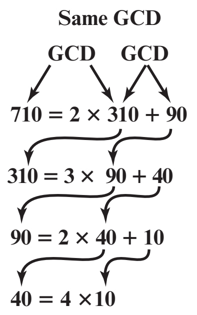


Figure 2.3 Euclidean Algorithm Example: gcd(710, 310)

Let us now return to Equation (2.2) and assume that $r_1 \neq 0$. Because $b > r_1$, we can divide b by r_1 and apply the division algorithm to obtain:

$$b = q_2 r_1 + r_2 \qquad 0 \leq r_2 < r_1$$

As before, if $r_2 = 0$, then $d = r_1$ and if $r_2 \neq 0$, then $d = \gcd(r_1, r_2)$. Note that the remainders form a descending series of nonnegative values and so must terminate when the remainder is zero. This happens, say, at the $(n + 1)$th stage where r_{n-1} is divided by r_n. The result is the following system of equations:

$$\begin{array}{ll} a = q_1 b + r_1 & 0 < r_1 < b \\ b = q_2 r_1 + r_2 & 0 < r_2 < r_1 \\ r_1 = q_3 r_2 + r_3 & 0 < r_3 < r_2 \\ \quad\vdots & \quad\vdots \\ r_{n-2} = q_n r_{n-1} + r_n & 0 < r_n < r_{n-1} \\ r_{n-1} = q_{n+1} r_n + 0 & \\ d = \gcd(a, b) = r_n & \end{array} \qquad \textbf{(2.3)}$$

At each iteration, we have $d = \gcd(r_i, r_{i+1})$ until finally $d = \gcd(r_n, 0) = r_n$. Thus, we can find the greatest common divisor of two integers by repetitive application of the division algorithm. This scheme is known as the Euclidean algorithm. Figure 2.3 illustrates a simple example.

We have essentially argued from the top down that the final result is the $\gcd(a, b)$. We can also argue from the bottom up. The first step is to show that r_n divides a and b. It follows from the last division in Equation (2.3) that r_n divides r_{n-1}. The next to last division shows that r_n divides r_{n-2} because it divides both terms on the right. Successively, one sees that r_n divides all r_i's and finally a and b. It remains to show that r_n is the largest divisor that divides a and b. If we take any arbitrary integer that divides a and b, it must also divide r_1, as explained previously. We can follow the sequence of equations in Equation (2.3) down and show that c must divide all r_i's. Therefore c must divide r_n, so that $r_n = \gcd(a, b)$.

Let us now look at an example with relatively large numbers to see the power of this algorithm:

To find $d = \gcd(a, b) = \gcd(1160718174, 316258250)$		
$a = q_1 b + r_1$	$1160718174 = 3 \times 316258250 + 211943424$	$d = \gcd(316258250, 211943424)$
$b = q_2 r_1 + r_2$	$316258250 = 1 \times 211943424 + 104314826$	$d = \gcd(211943424, 104314826)$
$r_1 = q_3 r_2 + r_3$	$211943424 = 2 \times 104314826 + 3313772$	$d = \gcd(104314826, 3313772)$
$r_2 = q_4 r_3 + r_4$	$104314826 = 31 \times 3313772 + 1587894$	$d = \gcd(3313772, 1587894)$
$r_3 = q_5 r_4 + r_5$	$3313772 = 2 \times 1587894 + 137984$	$d = \gcd(1587894, 137984)$
$r_4 = q_6 r_5 + r_6$	$1587894 = 11 \times 137984 + 70070$	$d = \gcd(137984, 70070)$
$r_5 = q_7 r_6 + r_7$	$137984 = 1 \times 70070 + 67914$	$d = \gcd(70070, 67914)$
$r_6 = q_8 r_7 + r_8$	$70070 = 1 \times 67914 + 2156$	$d = \gcd(67914, 2156)$
$r_7 = q_9 r_8 + r_9$	$67914 = 31 \times 2156 + 1078$	$d = \gcd(2156, 1078)$
$r_8 = q_{10} r_9 + r_{10}$	$2156 = 2 \times 1078 + 0$	$d = \gcd(1078, 0) = 1078$
Therefore, $d = \gcd(1160718174, 316258250) = 1078$		

In this example, we begin by dividing 1160718174 by 316258250, which gives 3 with a remainder of 211943424. Next we take 316258250 and divide it by 211943424. The process continues until we get a remainder of 0, yielding a result of 1078.

It will be helpful in what follows to recast the above computation in tabular form. For every step of the iteration, we have $r_{i-2} = q_i r_{i-1} + r_i$, where r_{i-2} is the dividend, r_{i-1} is the divisor, q_i is the quotient, and r_i is the remainder. Table 2.1 summarizes the results.

Table 2.1 Euclidean Algorithm Example

Dividend	Divisor	Quotient	Remainder
$a = 1160718174$	$b = 316258250$	$q_1 = 3$	$r_1 = 211943424$
$b = 316258250$	$r_1 = 211943434$	$q_2 = 1$	$r_2 = 104314826$
$r_1 = 211943424$	$r_2 = 104314826$	$q_3 = 2$	$r_3 = 3313772$
$r_2 = 104314826$	$r_3 = 3313772$	$q_4 = 31$	$r_4 = 1587894$
$r_3 = 3313772$	$r_4 = 1587894$	$q_5 = 2$	$r_5 = 137984$
$r_4 = 1587894$	$r_5 = 137984$	$q_6 = 11$	$r_6 = 70070$
$r_5 = 137984$	$r_6 = 70070$	$q_7 = 1$	$r_7 = 67914$
$r_6 = 70070$	$r_7 = 67914$	$q_8 = 1$	$r_8 = 2156$
$r_7 = 67914$	$r_8 = 2156$	$q_9 = 31$	$r_9 = 1078$
$r_8 = 2156$	$r_9 = 1078$	$q_{10} = 2$	$r_{10} = 0$

2.3 MODULAR ARITHMETIC

The Modulus

If a is an integer and n is a positive integer, we define $a \bmod n$ to be the remainder when a is divided by n. The integer n is called the **modulus**. Thus, for any integer a, we can rewrite Equation (2.1) as follows:

$$a = qn + r \qquad 0 \leq r < n; q = \lfloor a/n \rfloor$$
$$a = \lfloor a/n \rfloor \times n + (a \bmod n)$$

$$\boxed{11 \bmod 7 = 4; \qquad -11 \bmod 7 = 3}$$

Two integers a and b are said to be **congruent modulo n**, if $(a \bmod n) = (b \bmod n)$. This is written as $a \equiv b \pmod{n}$.[2]

$$\boxed{73 \equiv 4 \pmod{23}; \qquad 21 \equiv -9 \pmod{10}}$$

Note that if $a \equiv 0 \pmod{n}$, then $n | a$.

Properties of Congruences

Congruences have the following properties:

1. $a \equiv b \pmod{n}$ if $n | (a - b)$.
2. $a \equiv b \pmod{n}$ implies $b \equiv a \pmod{n}$.
3. $a \equiv b \pmod{n}$ and $b \equiv c \pmod{n}$ imply $a \equiv c \pmod{n}$.

To demonstrate the first point, if $n | (a - b)$, then $(a - b) = kn$ for some k. So we can write $a = b + kn$. Therefore, $(a \bmod n) =$ (remainder when $b + kn$ is divided by n) $=$ (remainder when b is divided by n) $= (b \bmod n)$.

$23 \equiv 8 \pmod{5}$	because	$23 - 8 = 15 = 5 \times 3$
$-11 \equiv 5 \pmod{8}$	because	$-11 - 5 = -16 = 8 \times (-2)$
$81 \equiv 0 \pmod{27}$	because	$81 - 0 = 81 = 27 \times 3$

The remaining points are as easily proved.

[2]We have just used the operator *mod* in two different ways: first as a **binary operator** that produces a remainder, as in the expression $a \bmod b$; second as a **congruence relation** that shows the equivalence of two integers, as in the expression $a \equiv b \pmod{n}$. See Appendix 2A for a discussion.

Modular Arithmetic Operations

Note that, by definition (Figure 2.1), the (mod n) operator maps all integers into the set of integers $\{0, 1, \ldots, (n - 1)\}$. This suggests the question: Can we perform arithmetic operations within the confines of this set? It turns out that we can; this technique is known as **modular arithmetic**.

Modular arithmetic exhibits the following properties:

1. $[(a \bmod n) + (b \bmod n)] \bmod n = (a + b) \bmod n$
2. $[(a \bmod n) - (b \bmod n)] \bmod n = (a - b) \bmod n$
3. $[(a \bmod n) \times (b \bmod n)] \bmod n = (a \times b) \bmod n$

We demonstrate the first property. Define $(a \bmod n) = r_a$ and $(b \bmod n) = r_b$. Then we can write $a = r_a + jn$ for some integer j and $b = r_b + kn$ for some integer k. Then

$$\begin{aligned}(a + b) \bmod n &= (r_a + jn + r_b + kn) \bmod n \\ &= (r_a + r_b + (k + j)n) \bmod n \\ &= (r_a + r_b) \bmod n \\ &= [(a \bmod n) + (b \bmod n)] \bmod n\end{aligned}$$

The remaining properties are proven as easily. Here are examples of the three properties:

$11 \bmod 8 = 3$; $15 \bmod 8 = 7$
$[(11 \bmod 8) + (15 \bmod 8)] \bmod 8 = 10 \bmod 8 = 2$
$(11 + 15) \bmod 8 = 26 \bmod 8 = 2$
$[(11 \bmod 8) - (15 \bmod 8)] \bmod 8 = -4 \bmod 8 = 4$
$(11 - 15) \bmod 8 = -4 \bmod 8 = 4$
$[(11 \bmod 8) \times (15 \bmod 8)] \bmod 8 = 21 \bmod 8 = 5$
$(11 \times 15) \bmod 8 = 165 \bmod 8 = 5$

Exponentiation is performed by repeated multiplication, as in ordinary arithmetic.

To find $11^7 \bmod 13$, we can proceed as follows:

$11^2 = 121 \equiv 4 \pmod{13}$
$11^4 = (11^2)^2 \equiv 4^2 \equiv 3 \pmod{13}$
$11^7 = 11 \times 11^2 \times 11^4$
$11^7 \equiv 11 \times 4 \times 3 \equiv 132 \equiv 2 \pmod{13}$

Thus, the rules for ordinary arithmetic involving addition, subtraction, and multiplication carry over into modular arithmetic.

Table 2.2 provides an illustration of modular addition and multiplication modulo 8. Looking at addition, the results are straightforward, and there is a regular pattern to the matrix. Both matrices are symmetric about the main diagonal in conformance to the **commutative** property of addition and multiplication. As in ordinary addition, there is an additive inverse, or negative, to each integer in modular arithmetic. In this case, the negative of an integer x is the integer y such that $(x + y) \bmod 8 = 0$. To find the additive inverse of an integer in the left-hand column, scan across the corresponding row of the matrix to find the value 0; the integer at the top of that column is the additive inverse; thus, $(2 + 6) \bmod 8 = 0$. Similarly, the entries in the multiplication table are straightforward. In modular arithmetic mod 8, the multiplicative inverse of x is the integer y such that $(x \times y) \bmod 8 = 1 \bmod 8$. Now, to find the multiplicative inverse of an integer from the multiplication table, scan across the matrix in the row for that integer to find the value 1; the integer at the top of that column is the multiplicative inverse; thus, $(3 \times 3) \bmod 8 = 1$. Note that not all integers mod 8 have a multiplicative inverse; more about that later.

Properties of Modular Arithmetic

Define the set Z_n as the set of nonnegative integers less than n:

$$Z_n = \{0, 1, \ldots, (n - 1)\}$$

Table 2.2 Arithmetic Modulo 8

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	0
2	2	3	4	5	6	7	0	1
3	3	4	5	6	7	0	1	2
4	4	5	6	7	0	1	2	3
5	5	6	7	0	1	2	3	4
6	6	7	0	1	2	3	4	5
7	7	0	1	2	3	4	5	6

(a) Addition modulo 8

×	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	0	2	4	6
3	0	3	6	1	4	7	2	5
4	0	4	0	4	0	4	0	4
5	0	5	2	7	4	1	6	3
6	0	6	4	2	0	6	4	2
7	0	7	6	5	4	3	2	1

(b) Multiplication modulo 8

w	$-w$	w^{-1}
0	0	—
1	7	1
2	6	—
3	5	3
4	4	—
5	3	5
6	2	—
7	1	7

(c) Additive and multiplicative inverse modulo 8

This is referred to as the **set of residues**, or **residue classes** (mod n). To be more precise, each integer in Z_n represents a residue class. We can label the residue classes (mod n) as $[0], [1], [2], \ldots, [n-1]$, where

$$[r] = \{a: a \text{ is an integer}, a \equiv r \pmod{n}\}$$

The residue classes (mod 4) are

$$[0] = \{\ldots, -16, -12, -8, -4, 0, 4, 8, 12, 16, \ldots\}$$
$$[1] = \{\ldots, -15, -11, -7, -3, 1, 5, 9, 13, 17, \ldots\}$$
$$[2] = \{\ldots, -14, -10, -6, -2, 2, 6, 10, 14, 18, \ldots\}$$
$$[3] = \{\ldots, -13, -9, -5, -1, 3, 7, 11, 15, 19, \ldots\}$$

Of all the integers in a residue class, the smallest nonnegative integer is the one used to represent the residue class. Finding the smallest nonnegative integer to which k is congruent modulo n is called **reducing k modulo n**.

If we perform modular arithmetic within Z_n, the properties shown in Table 2.3 hold for integers in Z_n. We show in Chapter 5 that this implies that Z_n is a commutative ring with a multiplicative **identity element**.

There is one peculiarity of modular arithmetic that sets it apart from ordinary arithmetic. First, observe that (as in ordinary arithmetic) we can write the following:

$$\textbf{if } (a + b) \equiv (a + c) \pmod{n} \textbf{ then } b \equiv c \pmod{n} \tag{2.4}$$

$$(5 + 23) \equiv (5 + 7) \pmod{8}; 23 \equiv 7 \pmod{8}$$

Equation (2.4) is consistent with the existence of an additive inverse. Adding the additive inverse of a to both sides of Equation (2.4), we have

$$((-a) + a + b) \equiv ((-a) + a + c) \pmod{n}$$
$$b \equiv c \pmod{n}$$

Table 2.3 Properties of Modular Arithmetic for Integers in Z_n

Property	Expression
Commutative Laws	$(w + x) \bmod n = (x + w) \bmod n$ $(w \times x) \bmod n = (x \times w) \bmod n$
Associative Laws	$[(w + x) + y] \bmod n = [w + (x + y)] \bmod n$ $[(w \times x) \times y] \bmod n = [w \times (x \times y)] \bmod n$
Distributive Law	$[w \times (x + y)] \bmod n = [(w \times x) + (w \times y)] \bmod n$
Identities	$(0 + w) \bmod n = w \bmod n$ $(1 \times w) \bmod n = w \bmod n$
Additive Inverse $(-w)$	For each $w \in Z_n$, there exists a z such that $w + z \equiv 0 \bmod n$

However, the following statement is true only with the attached condition:

if $(a \times b) \equiv (a \times c)(\text{mod } n)$ **then** $b \equiv c(\text{mod } n)$ **if** a is relatively prime to n **(2.5)**

Recall that two integers are **relatively prime** if their only common positive integer factor is 1. Similar to the case of Equation (2.4), we can say that Equation (2.5) is consistent with the existence of a multiplicative inverse. Applying the multiplicative inverse of a to both sides of Equation (2.5), we have

$$((a^{-1})ab) \equiv ((a^{-1})ac)(\text{mod } n)$$
$$b \equiv c(\text{mod } n)$$

To see this, consider an example in which the condition of Equation (2.5) does not hold. The integers 6 and 8 are not relatively prime, since they have the common factor 2. We have the following:

$$6 \times 3 = 18 \equiv 2(\text{mod } 8)$$
$$6 \times 7 = 42 \equiv 2(\text{mod } 8)$$

Yet $3 \not\equiv 7$ (mod 8).

The reason for this strange result is that for any general modulus n, a multiplier a that is applied in turn to the integers 0 through $(n - 1)$ will fail to produce a complete set of residues if a and n have any factors in common.

With $a = 6$ and $n = 8$,

Z_8	0	1	2	3	4	5	6	7
Multiply by 6	0	6	12	18	24	30	36	42
Residues	0	6	4	2	0	6	4	2

Because we do not have a complete set of residues when multiplying by 6, more than one integer in Z_8 maps into the same residue. Specifically, $6 \times 0 \text{ mod } 8 = 6 \times 4 \text{ mod } 8$; $6 \times 1 \text{ mod } 8 = 6 \times 5 \text{ mod } 8$; and so on. Because this is a many-to-one mapping, there is not a unique inverse to the multiply operation.

However, if we take $a = 5$ and $n = 8$, whose only common factor is 1,

Z_8	0	1	2	3	4	5	6	7
Multiply by 5	0	5	10	15	20	25	30	35
Residues	0	5	2	7	4	1	6	3

The line of residues contains all the integers in Z_8, in a different order.

In general, an integer has a multiplicative inverse in Z_n if and only if that integer is relatively prime to n. Table 2.2c shows that the integers 1, 3, 5, and 7 have a multiplicative inverse in Z_8; but 2, 4, and 6 do not.

Euclidean Algorithm Revisited

The Euclidean algorithm can be based on the following theorem: For any integers a, b, with $a \geq b \geq 0$,

$$\gcd(a, b) = \gcd(b, a \bmod b) \tag{2.6}$$

$$\boxed{\gcd(55, 22) = \gcd(22, 55 \bmod 22) = \gcd(22, 11) = 11}$$

To see that Equation (2.6) works, let $d = \gcd(a, b)$. Then, by the definition of gcd, $d|a$ and $d|b$. For any positive integer b, we can express a as

$$a = kb + r \equiv r \pmod{b}$$
$$a \bmod b = r$$

with k, r integers. Therefore, $(a \bmod b) = a - kb$ for some integer k. But because $d|b$, it also divides kb. We also have $d|a$. Therefore, $d|(a \bmod b)$. This shows that d is a common divisor of b and $(a \bmod b)$. Conversely, if d is a common divisor of b and $(a \bmod b)$, then $d|kb$ and thus $d|[kb + (a \bmod b)]$, which is equivalent to $d|a$. Thus, the set of common divisors of a and b is equal to the set of common divisors of b and $(a \bmod b)$. Therefore, the gcd of one pair is the same as the gcd of the other pair, proving the theorem.

Equation (2.6) can be used repetitively to determine the greatest common divisor.

$$\boxed{\begin{array}{l}\gcd(18, 12) = \gcd(12, 6) = \gcd(6, 0) = 6 \\ \gcd(11, 10) = \gcd(10, 1) = \gcd(1, 0) = 1\end{array}}$$

This is the same scheme shown in Equation (2.3), which can be rewritten in the following way.

Euclidean Algorithm	
Calculate	**Which satisfies**
$r_1 = a \bmod b$	$a = q_1 b + r_1$
$r_2 = b \bmod r_1$	$b = q_2 r_1 + r_2$
$r_3 = r_1 \bmod r_2$	$r_1 = q_3 r_2 + r_3$
• • •	• • •
$r_n = r_{n-2} \bmod r_{n-1}$	$r_{n-2} = q_n r_{n-1} + r_n$
$r_{n+1} = r_{n-1} \bmod r_n = 0$	$r_{n-1} = q_{n+1} r_n + 0$ $d = \gcd(a, b) = r_n$

We can define the Euclidean algorithm concisely as the following recursive function.

```
Euclid(a,b)
    if (b=0) then return a;
    else return Euclid(b, a mod b);
```

The Extended Euclidean Algorithm

We now proceed to look at an extension to the Euclidean algorithm that will be important for later computations in the area of finite fields and in encryption algorithms, such as RSA. For given integers a and b, the extended Euclidean algorithm not only calculates the greatest common divisor d but also two additional integers x and y that satisfy the following equation.

$$ax + by = d = \gcd(a, b) \tag{2.7}$$

It should be clear that x and y will have opposite signs. Before examining the algorithm, let us look at some of the values of x and y when $a = 42$ and $b = 30$. Note that $\gcd(42, 30) = 6$. Here is a partial table of values[3] for $42x + 30y$.

x / *y*	**−3**	**−2**	**−1**	**0**	**1**	**2**	**3**
−3	−216	−174	−132	−90	−48	−6	36
−2	−186	−144	−102	−60	−18	24	66
−1	−156	−114	−72	−30	12	54	96
0	−126	−84	−42	0	42	84	126
1	−96	−54	−12	30	72	114	156
2	−66	−24	18	60	102	144	186
3	−36	6	48	90	132	174	216

Observe that all of the entries are divisible by 6. This is not surprising, because both 42 and 30 are divisible by 6, so every number of the form $42x + 30y = 6(7x + 5y)$ is a multiple of 6. Note also that $\gcd(42, 30) = 6$ appears in the table. In general, it can be shown that for given integers a and b, the smallest positive value of $ax + by$ is equal to $\gcd(a, b)$.

Now let us show how to extend the Euclidean algorithm to determine (x, y, d) given a and b. We again go through the sequence of divisions indicated in Equation (2.3), and we assume that at each step i we can find integers x_i and y_i that satisfy $r_i = ax_i + by_i$. We end up with the following sequence.

$$\begin{array}{rlrl}
a &= q_1 b + r_1 & r_1 &= ax_1 + by_1 \\
b &= q_2 r_1 + r_2 & r_2 &= ax_2 + by_2 \\
r_1 &= q_3 r_2 + r_3 & r_3 &= ax_3 + by_3 \\
&\vdots & &\vdots \\
r_{n-2} &= q_n r_{n-1} + r_n & r_n &= ax_n + by_n \\
r_{n-1} &= q_{n+1} r_n + 0 & &
\end{array}$$

[3]This example is taken from [SILV06].

Now, observe that we can rearrange terms to write

$$r_i = r_{i-2} - r_{i-1}q_i \tag{2.8}$$

Also, in rows $i - 1$ and $i - 2$, we find the values

$$r_{i-2} = ax_{i-2} + by_{i-2} \quad \text{and} \quad r_{i-1} = ax_{i-1} + by_{i-1}$$

Substituting into Equation (2.8), we have

$$\begin{aligned} r_i &= (ax_{i-2} + by_{i-2}) - (ax_{i-1} + by_{i-1})q_i \\ &= a(x_{i-2} - q_i x_{i-1}) + b(y_{i-2} - q_i y_{i-1}) \end{aligned}$$

But we have already assumed that $r_i = ax_i + by_i$. Therefore,

$$x_i = x_{i-2} - q_i x_{i-1} \quad \text{and} \quad y_i = y_{i-2} - q_i y_{i-1}$$

We now summarize the calculations:

Extended Euclidean Algorithm

Calculate	Which satisfies	Calculate	Which satisfies
$r_{-1} = a$		$x_{-1} = 1; y_{-1} = 0$	$a = ax_{-1} + by_{-1}$
$r_0 = b$		$x_0 = 0; y_0 = 1$	$b = ax_0 + by_0$
$r_1 = a \bmod b$ $q_1 = \lfloor a/b \rfloor$	$a = q_1 b + r_1$	$x_1 = x_{-1} - q_1 x_0 = 1$ $y_1 = y_{-1} - q_1 y_0 = -q_1$	$r_1 = ax_1 + by_1$
$r_2 = b \bmod r_1$ $q_2 = \lfloor b/r_1 \rfloor$	$b = q_2 r_1 + r_2$	$x_2 = x_0 - q_2 x_1$ $y_2 = y_0 - q_2 y_1$	$r_2 = ax_2 + by_2$
$r_3 = r_1 \bmod r_2$ $q_3 = \lfloor r_1/r_2 \rfloor$	$r_1 = q_3 r_2 + r_3$	$x_3 = x_1 - q_3 x_2$ $y_3 = y_1 - q_3 y_2$	$r_3 = ax_3 + by_3$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$r_n = r_{n-2} \bmod r_{n-1}$ $q_n = \lfloor r_{n-2}/r_{n-1} \rfloor$	$r_{n-2} = q_n r_{n-1} + r_n$	$x_n = x_{n-2} - q_n x_{n-1}$ $y_n = y_{n-2} - q_n y_{n-1}$	$r_n = ax_n + by_n$
$r_{n+1} = r_{n-1} \bmod r_n = 0$ $q_{n+1} = \lfloor r_{n-1}/r_n \rfloor$	$r_{n-1} = q_{n+1} r_n + 0$		$d = \gcd(a, b) = r_n$ $x = x_n; y = y_n$

We need to make several additional comments here. In each row, we calculate a new remainder r_i based on the remainders of the previous two rows, namely r_{i-1} and r_{i-2}. To start the algorithm, we need values for r_0 and r_{-1}, which are just a and b. It is then straightforward to determine the required values for x_{-1}, y_{-1}, x_0, and y_0.

We know from the original Euclidean algorithm that the process ends with a remainder of zero and that the greatest common divisor of a and b is $d = \gcd(a, b) = r_n$. But we also have determined that $d = r_n = ax_n + by_n$. Therefore, in Equation (2.7), $x = x_n$ and $y = y_n$.

As an example, let us use $a = 1759$ and $b = 550$ and solve for $1759x + 550y = \gcd(1759, 550)$. The results are shown in Table 2.4. Thus, we have $1759 \times (-111) + 550 \times 355 = -195249 + 195250 = 1$.

Table 2.4 Extended Euclidean Algorithm Example

i	r_i	q_i	x_i	y_i
−1	1759		1	0
0	550		0	1
1	109	3	1	−3
2	5	5	−5	16
3	4	21	106	−339
4	1	1	−111	355
5	0	4		

Result: $d = 1; x = -111; y = 355$

2.4 PRIME NUMBERS[4]

A central concern of number theory is the study of prime numbers. Indeed, whole books have been written on the subject (e.g., [CRAN01], [RIBE96]). In this section, we provide an overview relevant to the concerns of this book.

An integer $p > 1$ is a **prime number** if and only if its only divisors[5] are ± 1 and $\pm p$. All numbers other than ± 1 and the prime numbers are **composite numbers**. In other words, composite numbers are those which are the product of at least two prime numbers. Prime numbers play a critical role in number theory and in the techniques discussed in this chapter. Table 2.5 shows the primes less than 2000. Note the way the primes are distributed. In particular, note the number of primes in each range of 100 numbers.

Any integer $a > 1$ can be factored in a unique way as

$$a = p_1^{a_1} \times p_2^{a_2} \times \cdots \times p_t^{a_t} \tag{2.9}$$

where $p_1 < p_2 < \ldots < p_t$ are prime numbers and where each a_i is a positive integer. This is known as the fundamental theorem of arithmetic; a proof can be found in any text on number theory.

$$\begin{aligned} 91 &= 7 \times 13 \\ 3600 &= 2^4 \times 3^2 \times 5^2 \\ 11011 &= 7 \times 11^2 \times 13 \end{aligned}$$

It is useful for what follows to express Equation (2.9) another way. If P is the set of all prime numbers, then any positive integer a can be written uniquely in the following form:

$$a = \prod_{p \in \mathrm{P}} p^{a_p} \quad \text{where each } a_p \geq 0$$

[4]In this section, unless otherwise noted, we deal only with the nonnegative integers. The use of negative integers would introduce no essential differences.

[5]Recall from Section 2.1 that integer a is said to be a divisor of integer b if there is no remainder on division. Equivalently, we say that a divides b.

Table 2.5 Primes Under 2000

2	101	211	307	401	503	601	701	809	907	1009	1103	1201	1301	1409	1511	1601	1709	1801	1901
3	103	223	311	409	509	607	709	811	911	1013	1109	1213	1303	1423	1523	1607	1721	1811	1907
5	107	227	313	419	521	613	719	821	919	1019	1117	1217	1307	1427	1531	1609	1723	1823	1913
7	109	229	317	421	523	617	727	823	929	1021	1123	1223	1319	1429	1543	1613	1733	1831	1931
11	113	233	331	431	541	619	733	827	937	1031	1129	1229	1321	1433	1549	1619	1741	1847	1933
13	127	239	337	433	547	631	739	829	941	1033	1151	1231	1327	1439	1553	1621	1747	1861	1949
17	131	241	347	439	557	641	743	839	947	1039	1153	1237	1361	1447	1559	1627	1753	1867	1951
19	137	251	349	443	563	643	751	853	953	1049	1163	1249	1367	1451	1567	1637	1759	1871	1973
23	139	257	353	449	569	647	757	857	967	1051	1171	1259	1373	1453	1571	1657	1777	1873	1979
29	149	263	359	457	571	653	761	859	971	1061	1181	1277	1381	1459	1579	1663	1783	1877	1987
31	151	269	367	461	577	659	769	863	977	1063	1187	1279	1399	1471	1583	1667	1787	1879	1993
37	157	271	373	463	587	661	773	877	983	1069	1193	1283		1481	1597	1669	1789	1889	1997
41	163	277	379	467	593	673	787	881	991	1087		1289		1483		1693			1999
43	167	281	383	479	599	677	797	883	997	1091		1291		1487		1697			
47	173	283	389	487		683		887		1093		1297		1489		1699			
53	179	293	397	491		691				1097				1493					
59	181			499										1499					
61	191																		
67	193																		
71	197																		
73	199																		
79																			
83																			
89																			
97																			

The right-hand side is the product over all possible prime numbers p; for any particular value of a, most of the exponents a_p will be 0.

The value of any given positive integer can be specified by simply listing all the nonzero exponents in the foregoing formulation.

The integer 12 is represented by $\{a_2 = 2, a_3 = 1\}$.
The integer 18 is represented by $\{a_2 = 1, a_3 = 2\}$.
The integer 91 is represented by $\{a_7 = 1, a_{13} = 1\}$.

Multiplication of two numbers is equivalent to adding the corresponding exponents. Given $a = \prod_{p \in \mathrm{P}} p^{a_p}$, $b = \prod_{p \in \mathrm{P}} p^{b_p}$. Define $k = ab$. We know that the integer k can be expressed as the product of powers of primes: $k = \prod_{p \in \mathrm{P}} p^{k_p}$. It follows that $k_p = a_p + b_p$ for all $p \in \mathrm{P}$.

$k = 12 \times 18 = (2^2 \times 3) \times (2 \times 3^2) = 216$
$k_2 = 2 + 1 = 3; k_3 = 1 + 2 = 3$
$216 = 2^3 \times 3^3 = 8 \times 27$

What does it mean, in terms of the prime factors of a and b, to say that a divides b? Any integer of the form p^n can be divided only by an integer that is of a lesser or equal power of the same prime number, p^j with $j \leq n$. Thus, we can say the following.

Given

$$a = \prod_{p \in \mathrm{P}} p^{a_p}, b = \prod_{p \in \mathrm{P}} p^{b_p}$$

If $a|b$, then $a_p \leq b_p$ for all p.

$a = 12; b = 36; 12|36$
$12 = 2^2 \times 3; 36 = 2^2 \times 3^2$
$a_2 = 2 = b_2$
$a_3 = 1 \leq 2 = b_3$
Thus, the inequality $a_p \leq b_p$ is satisfied for all prime numbers.

It is easy to determine the greatest common divisor of two positive integers if we express each integer as the product of primes.

$$\boxed{\begin{aligned} 300 &= 2^2 \times 3^1 \times 5^2 \\ 18 &= 2^1 \times 3^2 \\ \gcd(18{,}300) &= 2^1 \times 3^1 \times 5^0 = 6 \end{aligned}}$$

The following relationship always holds:

$$\text{If } k = \gcd(a, b), \text{ then } k_p = \min(a_p, b_p) \text{ for all } p.$$

Determining the prime factors of a large number is no easy task, so the preceding relationship does not directly lead to a practical method of calculating the greatest common divisor.

2.5 FERMAT'S AND EULER'S THEOREMS

Two theorems that play important roles in public-key cryptography are Fermat's theorem and Euler's theorem.

Fermat's Theorem[6]

Fermat's theorem states the following: If p is prime and a is a positive integer not divisible by p, then

$$a^{p-1} \equiv 1 \pmod{p} \tag{2.10}$$

Proof: Consider the set of positive integers less than p: $\{1, 2, \ldots, p-1\}$ and multiply each element by a, modulo p, to get the set $X = \{a \bmod p, 2a \bmod p, \ldots, (p-1)a \bmod p\}$. None of the elements of X is equal to zero because p does not divide a. Furthermore, no two of the integers in X are equal. To see this, assume that $ja \equiv ka(\bmod\, p))$, where $1 \le j < k \le p - 1$. Because a is relatively prime[7] to p, we can eliminate a from both sides of the equation [see Equation (2.5)] resulting in $j \equiv k(\bmod\, p)$. This last equality is impossible, because j and k are both positive integers less than p. Therefore, we know that the $(p-1)$ elements of X are all positive integers with no two elements equal. We can conclude the X consists of the set of integers $\{1, 2, \ldots, p-1\}$ in some order. Multiplying the numbers in both sets (p and X) and taking the result mod p yields

$$a \times 2a \times \cdots \times (p-1)a \equiv [(1 \times 2 \times \cdots \times (p-1)](\bmod\, p)$$
$$a^{p-1}(p-1)! \equiv (p-1)! \pmod{p}$$

We can cancel the $(p-1)!$ term because it is relatively prime to p [see Equation (2.5)]. This yields Equation (2.10), which completes the proof.

[6]This is sometimes referred to as Fermat's little theorem.

[7]Recall from Section 2.2 that two numbers are relatively prime if they have no prime factors in common; that is, their only common divisor is 1. This is equivalent to saying that two numbers are relatively prime if their greatest common divisor is 1.

$a = 7, p = 19$
$7^2 = 49 \equiv 11 \pmod{19}$
$7^4 \equiv 121 \equiv 7 \pmod{19}$
$7^8 \equiv 49 \equiv 11 \pmod{19}$
$7^{16} \equiv 121 \equiv 7 \pmod{19}$
$a^{p-1} = 7^{18} = 7^{16} \times 7^2 \equiv 7 \times 11 \equiv 1 \pmod{19}$

An alternative form of Fermat's theorem is also useful: If p is prime and a is a positive integer, then

$$a^p \equiv a(\bmod p) \tag{2.11}$$

Note that the first form of the theorem [Equation (2.10)] requires that a be relatively prime to p, but this Equation (2.11) does not.

$p = 5, a = 3$	$a^p = 3^5 = 243 \equiv 3(\bmod 5) = a(\bmod p)$
$p = 5, a = 10$	$a^p = 10^5 = 100000 \equiv 10(\bmod 5) \equiv 0(\bmod 5) = a(\bmod p)$

Euler's Totient Function

Before presenting Euler's theorem, we need to introduce an important quantity in number theory, referred to as Euler's totient function. This function, written $\phi(n)$, is defined as the number of positive integers less than n and relatively prime to n. By convention, $\phi(1) = 1$.

Determine $\phi(37)$ and $\phi(35)$.

Because 37 is prime, all of the positive integers from 1 through 36 are relatively prime to 37. Thus $\phi(37) = 36$.

To determine $\phi(35)$, we list all of the positive integers less than 35 that are relatively prime to it:

1, 2, 3, 4, 6, 8, 9, 11, 12, 13, 16, 17, 18
19, 22, 23, 24, 26, 27, 29, 31, 32, 33, 34

There are 24 numbers on the list, so $\phi(35) = 24$.

Table 2.6 lists the first 30 values of $\phi(n)$. The value $\phi(1)$ is without meaning but is defined to have the value 1.

It should be clear that, for a prime number p,

$$\phi(p) = p - 1$$

Now suppose that we have two prime numbers p and q with $p \neq q$. Then we can show that, for $n = pq$,

Table 2.6 Some Values of Euler's Totient Function $\phi(n)$

n	$\phi(n)$
1	1
2	1
3	2
4	2
5	4
6	2
7	6
8	4
9	6
10	4
11	10
12	4
13	12
14	6
15	8
16	8
17	16
18	6
19	18
20	8
21	12
22	10
23	22
24	8
25	20
26	12
27	18
28	12
29	28
30	8

$$\phi(n) = \phi(pq) = \phi(p) \times \phi(q) = (p - 1) \times (q - 1)$$

To see that $\phi(n) = \phi(p) \times \phi(q)$, consider that the set of positive integers less than n is the set $\{1, \ldots, (pq - 1)\}$. The integers in this set that are not relatively prime to n are the set $\{p, 2p, \ldots, (q - 1)p\}$ and the set $\{q, 2q, \ldots, (p - 1)q\}$. To see this, consider that any integer that divides n must divide either of the prime numbers p or q. Therefore, any integer that does not contain either p or q as a factor is relatively prime to n. Further note that the two sets just listed are non-overlapping: Because p and q are prime, we can state that none of the integers in the first set can be written as a multiple of q, and none of the integers in the second set can be written as a multiple of p. Thus the total number of unique integers in the two sets is $(q - 1) + (p - 1)$. Accordingly,

$$\begin{aligned}
\phi(n) &= (pq - 1) - [(q - 1) + (p - 1)] \\
&= pq - (p + q) + 1 \\
&= (p - 1) \times (q - 1) \\
&= \phi(p) \times \phi(q)
\end{aligned}$$

$\phi(21) = \phi(3) \times \phi(7) = (3 - 1) \times (7 - 1) = 2 \times 6 = 12$
where the 12 integers are $\{1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20\}$.

Euler's Theorem

Euler's theorem states that for every a and n that are relatively prime:

$$a^{\phi(n)} \equiv 1 (\text{mod } n) \qquad \textbf{(2.12)}$$

Proof: Equation (2.12) is true if n is prime, because in that case, $\phi(n) = (n - 1)$ and Fermat's theorem holds. However, it also holds for any integer n. Recall that

$\phi(n)$ is the number of positive integers less than n that are relatively prime to n. Consider the set of such integers, labeled as

$$R = \{x_1, x_2, \ldots, x_{\phi(n)}\}$$

That is, each element x_i of R is a unique positive integer less than n with $\gcd(x_i, n) = 1$. Now multiply each element by a, modulo n:

$$S = \{(ax_1 \bmod n), (ax_2 \bmod n), \ldots, (ax_{\phi(n)} \bmod n)\}$$

The set S is a permutation[8] of R , by the following line of reasoning:

1. Because a is relatively prime to n and x_i is relatively prime to n, ax_i must also be relatively prime to n. Thus, all the members of S are integers that are less than n and that are relatively prime to n.
2. There are no duplicates in S. Refer to Equation (2.5). If $ax_i \bmod n = ax_j \bmod n$, then $x_i = x_j$.

Therefore,

$$\begin{aligned}
\prod_{i=1}^{\phi(n)} (ax_i \bmod n) &= \prod_{i=1}^{\phi(n)} x_i \\
\prod_{i=1}^{\phi(n)} ax_i &\equiv \prod_{i=1}^{\phi(n)} x_i \pmod n \\
a^{\phi(n)} \times \left[\prod_{i=1}^{\phi(n)} x_i\right] &\equiv \prod_{i=1}^{\phi(n)} x_i \pmod n \\
a^{\phi(n)} &\equiv 1 \pmod n
\end{aligned}$$

which completes the proof. This is the same line of reasoning applied to the proof of Fermat's theorem.

$a = 3; n = 10; \phi(10) = 4;$	$a^{\phi(n)} = 3^4 = 81 \equiv 1(\text{mod } 10) \equiv 1(\text{mod } n)$
$a = 2; n = 11; \phi(11) = 10;$	$a^{\phi(n)} = 2^{10} = 1024 \equiv 1(\text{mod } 11) \equiv 1(\text{mod } n)$

As is the case for Fermat's theorem, an alternative form of the theorem is also useful:

$$a^{\phi(n)+1} \equiv a(\text{mod } n) \qquad \textbf{(2.13)}$$

Again, similar to the case with Fermat's theorem, the first form of Euler's theorem [Equation (2.12)] requires that a be relatively prime to n, but this form does not. It is sufficient for Equation (2.13) that n is squarefree. An integer is squarefree if its prime decomposition contains no repeated factors.

[8]A permutation of a finite set of elements S is an ordered sequence of all the elements of S, with each element appearing exactly once.

2.6 TESTING FOR PRIMALITY

For many cryptographic algorithms, it is necessary to select one or more very large prime numbers at random. Thus, we are faced with the task of determining whether a given large number is prime. There is no simple yet efficient means of accomplishing this task.

In this section, we present one attractive and popular algorithm. You may be surprised to learn that this algorithm yields a number that is not necessarily a prime. However, the algorithm can yield a number that is almost certainly a prime. This will be explained presently. We also make reference to a deterministic algorithm for finding primes. The section closes with a discussion concerning the distribution of primes.

Miller–Rabin Algorithm[9]

The algorithm due to Miller and Rabin [MILL75, RABI80] is typically used to test a large number for primality. Before explaining the algorithm, we need some background. First, any positive odd integer $n \geq 3$ can be expressed as

$$n - 1 = 2^k q \qquad \text{with } k > 0, q \text{ odd}$$

To see this, note that $n - 1$ is an even integer. Then, divide $(n - 1)$ by 2 until the result is an odd number q, for a total of k divisions. If n is expressed as a binary number, then the result is achieved by shifting the number to the right until the rightmost digit is a 1, for a total of k shifts. We now develop two properties of prime numbers that we will need.

Two Properties of Prime Numbers The **first property** is stated as follows: If p is prime and a is a positive integer less than p, then $a^2 \bmod p = 1$ if and only if either $a \bmod p = 1$ or $a \bmod p = -1 \bmod p = p - 1$. By the rules of modular arithmetic $(a \bmod p)(a \bmod p) = a^2 \bmod p$. Thus, if either $a \bmod p = 1$ or $a \bmod p = -1$, then $a^2 \bmod p = 1$. Conversely, if $a^2 \bmod p = 1$, then $(a \bmod p)^2 = 1$, which is true only for $a \bmod p = 1$ or $a \bmod p = -1$.

The **second property** is stated as follows: Let p be a prime number greater than 2. We can then write $p - 1 = 2^k q$ with $k > 0$, q odd. Let a be any integer in the range $1 < a < p - 1$. Then one of the two following conditions is true.

1. a^q is congruent to 1 modulo p. That is, $a^q \bmod p = 1$, or equivalently, $a^q \equiv 1 (\bmod\ p)$.
2. One of the numbers $a^q, a^{2q}, a^{4q}, \ldots, a^{2^{k-1}q}$ is congruent to -1 modulo p. That is, there is some number j in the range $(1 \leq j \leq k)$ such that $a^{2^{j-1}q} \bmod p = -1 \bmod p = p - 1$ or equivalently, $a^{2^{j-1}q} \equiv -1 (\bmod\ p)$.

Proof: Fermat's theorem [Equation (2.10)] states that $a^{n-1} \equiv 1 (\bmod\ n)$ if n is prime. We have $p - 1 = 2^k q$. Thus, we know that $a^{p-1} \bmod p = a^{2^k q} \bmod p = 1$. Thus, if we look at the sequence of numbers

$$a^q \bmod p, a^{2q} \bmod p, a^{4q} \bmod p, \ldots, a^{2^{k-1}q} \bmod p, a^{2^k q} \bmod p \qquad \textbf{(2.14)}$$

[9]Also referred to in the literature as the Rabin-Miller algorithm, or the Rabin-Miller test, or the Miller–Rabin test.

we know that the last term in Equation (2.14) has value 1. Further, each number in the list is the square of the previous number. Therefore, one of the following possibilities must be true.

1. The first number on the list, and therefore all subsequent numbers on the list, equals 1.
2. Some number on the list does not equal 1, but its square mod p does equal 1. By virtue of the first property of prime numbers defined above, we know that the only number that satisfies this condition is $p - 1$. So, in this case, the list contains an element equal to $p - 1$.

This completes the proof.

Details of the Algorithm These considerations lead to the conclusion that, if n is prime, then either the first element in the list of residues, or remainders, $(a^q, a^{2q}, \ldots, a^{2^{k-1}q}, a^{2^k q})$ modulo n equals 1; or some element in the list equals $(n - 1)$; otherwise n is composite (i.e., not a prime). On the other hand, if the condition is met, that does not necessarily mean that n is prime. For example, if $n = 2047 = 23 \times 89$, then $n - 1 = 2 \times 1023$. We compute $2^{1023} \bmod 2047 = 1$, so that 2047 meets the condition but is not prime.

We can use the preceding property to devise a test for primality. The procedure TEST takes a candidate integer n as input and returns the result `composite` if n is definitely not a prime, and the result `inconclusive` if n may or may not be a prime.

```
TEST (n)
1. Find integers k, q, with k > 0, q odd, so that
   (n - 1 = 2k q);
2. Select a random integer a, 1 < a < n - 1;
3. if aq mod n = 1 then return("inconclusive");
4. for j = 0 to k - 1 do
5. if a^(2^j q) mod n = n - 1 then return("inconclusive");
6. return("composite");
```

Let us apply the test to the prime number $n = 29$. We have $(n - 1) = 28 = 2^2(7) = 2^k q$. First, let us try $a = 10$. We compute $10^7 \bmod 29 = 17$, which is neither 1 nor 28, so we continue the test. The next calculation finds that $(10^7)^2 \bmod 29 = 28$, and the test returns `inconclusive` (i.e., 29 may be prime). Let's try again with $a = 2$. We have the following calculations: $2^7 \bmod 29 = 12$; $2^{14} \bmod 29 = 28$; and the test again returns `inconclusive`. If we perform the test for all integers a in the range 1 through 28, we get the same `inconclusive` result, which is compatible with n being a prime number.

Now let us apply the test to the **composite number** $n = 13 \times 17 = 221$. Then $(n - 1) = 220 = 2^2(55) = 2^k q$. Let us try $a = 5$. Then we have $5^{55} \bmod 221 = 112$, which is neither 1 nor $220(5^{55})^2 \bmod 221 = 168$. Because we have used all values of j (i.e., $j = 0$ and $j = 1$) in line 4 of the TEST algorithm, the test returns `composite`, indicating that 221 is definitely a composite number. But suppose we had selected $a = 21$. Then we have $21^{55} \bmod 221 = 200$; $(21^{55})^2 \bmod 221 = 220$; and the test returns `inconclusive`, indicating that 221 may be prime. In fact, of the 218 integers from 2 through 219, four of these will return an inconclusive result, namely 21, 47, 174, and 200.

Repeated Use of the Miller–Rabin Algorithm How can we use the Miller–Rabin algorithm to determine with a high degree of confidence whether or not an integer is prime? It can be shown [KNUT98] that given an odd number n that is not prime and a randomly chosen integer, a with $1 < a < n - 1$, the probability that TEST will return `inconclusive` (i.e., fail to detect that n is not prime) is less than 1/4. Thus, if t different values of a are chosen, the probability that all of them will pass TEST (return inconclusive) for n is less than $(1/4)^t$. For example, for $t = 10$, the probability that a nonprime number will pass all ten tests is less than 10^{-6}. Thus, for a sufficiently large value of t, we can be confident that n is prime if Miller's test always returns `inconclusive`.

This gives us a basis for determining whether an odd integer n is prime with a reasonable degree of confidence. The procedure is as follows: Repeatedly invoke TEST (n) using randomly chosen values for a. If, at any point, TEST returns `composite`, then n is determined to be nonprime. If TEST continues to return `inconclusive` for t tests, then for a sufficiently large value of t, assume that n is prime.

A Deterministic Primality Algorithm

Prior to 2002, there was no known method of efficiently proving the primality of very large numbers. All of the algorithms in use, including the most popular (Miller–Rabin), produced a probabilistic result. In 2002 (announced in 2002, published in 2004), Agrawal, Kayal, and Saxena [AGRA04] developed a relatively simple deterministic algorithm that efficiently determines whether a given large number is a prime. The algorithm, known as the AKS algorithm, does not appear to be as efficient as the Miller–Rabin algorithm. Thus far, it has not supplanted this older, probabilistic technique.

Distribution of Primes

It is worth noting how many numbers are likely to be rejected before a prime number is found using the Miller–Rabin test, or any other test for primality. A result from number theory, known as the prime number theorem, states that the primes near n are spaced on the average one every ln (n) integers. Thus, on average, one would have to test on the order of ln(n) integers before a prime is found. Because all even integers can be immediately rejected, the correct figure is 0.5 ln(n). For example, if a prime on the order of magnitude of 2^{200} were sought, then about $0.5 \ln(2^{200}) = 69$ trials would be needed to find a prime. However, this figure is just an average. In some places along the number line, primes are closely packed, and in other places there are large gaps.

The two consecutive odd integers 1,000,000,000,061 and 1,000,000,000,063 are both prime. On the other hand, $1001! + 2, 1001! + 3, \dots, 1001! + 1000, 1001! + 1001$ is a sequence of 1000 consecutive composite integers.

2.7 THE CHINESE REMAINDER THEOREM

One of the most useful results of number theory is the Chinese remainder theorem (CRT).[10] In essence, the CRT says it is possible to reconstruct integers in a certain range from their residues modulo a set of pairwise relatively prime moduli.

The 10 integers in Z_{10}, that is the integers 0 through 9, can be reconstructed from their two residues modulo 2 and 5 (the relatively prime factors of 10). Say the known residues of a decimal digit x are $r_2 = 0$ and $r_5 = 3$; that is, $x \bmod 2 = 0$ and $x \bmod 5 = 3$. Therefore, x is an even integer in Z_{10} whose remainder, on division by 5, is 3. The unique solution is $x = 8$.

The CRT can be stated in several ways. We present here a formulation that is most useful from the point of view of this text. An alternative formulation is explored in Problem 2.33. Let

$$M = \prod_{i=1}^{k} m_i$$

where the m_i are pairwise relatively prime; that is, $\gcd(m_i, m_j) = 1$ for $1 \le i, j \le k$, and $i \ne j$. We can represent any integer A in Z_M by a k-tuple whose elements are in Z_{m_i} using the following correspondence:

$$A \leftrightarrow (a_1, a_2, \ldots , a_k) \qquad \textbf{(2.15)}$$

where $A \in Z_M$, $a_i \in Z_{m_i}$, and $a_i = A \bmod m_i$ for $1 \le i \le k$. The CRT makes two assertions.

1. The mapping of Equation (2.15) is a one-to-one correspondence (called a **bijection**) between Z_M and the Cartesian product $Z_{m_1} \times Z_{m_2} \times \ldots \times Z_{m_k}$. That is, for every integer A such that $0 \le A < M$, there is a unique k-tuple $(a_1, a_2, \ldots , a_k)$ with $0 \le a_i < m_i$ that represents it, and for every such k-tuple $(a_1, a_2, \ldots , a_k)$, there is a unique integer A in Z_M.
2. Operations performed on the elements of Z_M can be equivalently performed on the corresponding k-tuples by performing the operation independently in each coordinate position in the appropriate system.

Let us demonstrate the **first assertion**. The transformation from A to $(a_1, a_2, \ldots , a_k)$, is obviously unique; that is, each a_i is uniquely calculated as $a_i = A \bmod m_i$. Computing A from $(a_1, a_2, \ldots , a_k)$ can be done as follows. Let

[10]The CRT is so called because it is believed to have been discovered by the Chinese mathematician Sun-Tsu in around 100 A.D.

$M_i = M/m_i$ for $1 \le i \le k$. Note that $M_i = m_1 \times m_2 \times \ldots \times m_{i-1} \times m_{i+1} \times \ldots \times m_k$, so that $M_i \equiv 0 \pmod{m_j}$ for all $j \neq i$. Then let

$$c_i = M_i \times (M_i^{-1} \bmod m_i) \qquad \text{for } 1 \le i \le k \tag{2.16}$$

By the definition of M_i, it is relatively prime to m_i and therefore has a unique multiplicative inverse mod m_i. So Equation (2.16) is well defined and produces a unique value c_i. We can now compute

$$A \equiv \left(\sum_{i=1}^{k} a_i c_i\right) \pmod{M} \tag{2.17}$$

To show that the value of A produced by Equation (2.17) is correct, we must show that $a_i = A \bmod m_i$ for $1 \le i \le k$. Note that $c_j \equiv M_j \equiv 0 \pmod{m_i}$ if $j \neq i$, and that $c_i \equiv 1 \pmod{m_i}$. It follows that $a_i = A \bmod m_i$.

The **second assertion** of the CRT, concerning arithmetic operations, follows from the rules for modular arithmetic. That is, the second assertion can be stated as follows: If

$$A \leftrightarrow (a_1, a_2, \ldots, a_k)$$
$$B \leftrightarrow (b_1, b_2, \ldots, b_k)$$

then

$$(A + B) \bmod M \leftrightarrow ((a_1 + b_1) \bmod m_1, \ldots, (a_k + b_k) \bmod m_k)$$
$$(A - B) \bmod M \leftrightarrow ((a_1 - b_1) \bmod m_1, \ldots, (a_k - b_k) \bmod m_k)$$
$$(A \times B) \bmod M \leftrightarrow ((a_1 \times b_1) \bmod m_1, \ldots, (a_k \times b_k) \bmod m_k)$$

One of the useful features of the Chinese remainder theorem is that it provides a way to manipulate (potentially very large) numbers mod M in terms of tuples of smaller numbers. This can be useful when M is 150 digits or more. However, note that it is necessary to know beforehand the factorization of M.

To represent 973 mod 1813 as a pair of numbers mod 37 and 49, define

$$m_1 = 37$$
$$m_2 = 49$$
$$M = 1813$$
$$A = 973$$

We also have $M_1 = 49$ and $M_2 = 37$. Using the extended Euclidean algorithm, we compute $M_1^{-1} = 34 \bmod m_1$ and $M_2^{-1} = 4 \bmod m_2$. (Note that we only need to compute each M_i and each M_i^{-1} once.) Taking residues modulo 37 and 49, our representation of 973 is (11, 42), because 973 mod 37 = 11 and 973 mod 49 = 42.

Now suppose we want to add 678 to 973. What do we do to (11, 42)? First we compute $(678) \leftrightarrow (678 \bmod 37, 678 \bmod 49) = (12, 41)$. Then we add the tuples element-wise and reduce $(11 + 12 \bmod 37, 42 + 41 \bmod 49) = (23, 34)$. To verify that this has the correct effect, we compute

$$\begin{aligned}(23, 34) \leftrightarrow a_1M_1M_1^{-1} + a_2M_2M_2^{-1} \bmod M \\ &= [(23)(49)(34) + (34)(37)(4)] \bmod 1813 \\ &= 43350 \bmod 1813 \\ &= 1651\end{aligned}$$

and check that it is equal to $(973 + 678) \bmod 1813 = 1651$. Remember that in the above derivation, M_i^{-1} is the multiplicative inverse of M_1 modulo m_1 and M_2^{-1} is the multiplicative inverse of M_2 modulo m_2.

Suppose we want to multiply 1651 (mod 1813) by 73. We multiply (23, 34) by 73 and reduce to get $(23 \times 73 \bmod 37, 34 \times 73 \bmod 49) = (14, 32)$. It is easily verified that

$$\begin{aligned}(14, 32) &\leftrightarrow [(14)(49)(34) + (32)(37)(4)] \bmod 1813 \\ &= 865 \\ &= 1651 \times 73 \bmod 1813\end{aligned}$$

2.8 DISCRETE LOGARITHMS

Discrete logarithms are fundamental to a number of public-key algorithms, including Diffie–Hellman key exchange and the digital signature algorithm (DSA). This section provides a brief overview of discrete logarithms. For the interested reader, more detailed developments of this topic can be found in [ORE67] and [LEVE90].

The Powers of an Integer, Modulo *n*

Recall from Euler's theorem [Equation (2.12)] that, for every a and n that are relatively prime,

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

where $\phi(n)$, Euler's totient function, is the number of positive integers less than n and relatively prime to n. Now consider the more general expression:

$$a^m \equiv 1 \pmod{n} \tag{2.18}$$

If a and n are relatively prime, then there is at least one integer m that satisfies Equation (2.18), namely, $m = \phi(n)$. The least positive exponent m for which Equation (2.18) holds is referred to in several ways:

- The **order** of $a \pmod{n}$
- The exponent to which a belongs $\pmod{n}$
- The length of the period generated by a

To see this last point, consider the powers of 7, modulo 19:

$$\begin{aligned}
7^1 &\equiv & & 7 \pmod{19} \\
7^2 &= 49 = 2 \times 19 + 11 &\equiv\ & 11 \pmod{19} \\
7^3 &= 343 = 18 \times 19 + 1 &\equiv\ & 1 \pmod{19} \\
7^4 &= 2401 = 126 \times 19 + 7 &\equiv\ & 7 \pmod{19} \\
7^5 &= 16807 = 884 \times 19 + 11 &\equiv\ & 11 \pmod{19}
\end{aligned}$$

There is no point in continuing because the sequence is repeating. This can be proven by noting that $7^3 \equiv 1 (\text{mod } 19)$, and therefore, $7^{3+j} \equiv 7^3 7^j \equiv 7^j (\text{mod } 19)$, and hence, any two powers of 7 whose exponents differ by 3 (or a multiple of 3) are congruent to each other (mod 19). In other words, the sequence is periodic, and the length of the period is the smallest positive exponent m such that $7^m \equiv 1 (\text{mod } 19)$.

Table 2.7 shows all the powers of a, modulo 19 for all positive $a < 19$. The length of the sequence for each base value is indicated by shading. Note the following:

1. All sequences end in 1. This is consistent with the reasoning of the preceding few paragraphs.
2. The length of a sequence divides $\phi(19) = 18$. That is, an integral number of sequences occur in each row of the table.
3. Some of the sequences are of length 18. In this case, it is said that the base integer a generates (via powers) the set of nonzero integers modulo 19. Each such integer is called a primitive root of the modulus 19.

More generally, we can say that the highest possible exponent to which a number can belong (mod n) is $\phi(n)$. If a number is of this order, it is referred to as a **primitive root** of n. The importance of this notion is that if a is a primitive root of n, then its powers

$$a, a^2, \ldots, a^{\phi(n)}$$

are distinct (mod n) and are all relatively prime to n. In particular, for a prime number p, if a is a primitive root of p, then

$$a, a^2, \ldots, a^{p-1}$$

are distinct (mod p). For the prime number 19, its primitive roots are 2, 3, 10, 13, 14, and 15.

Not all integers have primitive roots. In fact, the only integers with primitive roots are those of the form 2, 4, p^α, and $2p^\alpha$, where p is any odd prime and α is a positive integer. The proof is not simple but can be found in many number theory books, including [ORE76].

Table 2.7 Powers of Integers, Modulo 19

a	a^2	a^3	a^4	a^5	a^6	a^7	a^8	a^9	a^{10}	a^{11}	a^{12}	a^{13}	a^{14}	a^{15}	a^{16}	a^{17}	a^{18}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1
3	9	8	5	15	7	2	6	18	16	10	11	14	4	12	17	13	1
4	16	7	9	17	11	6	5	1	4	16	7	9	17	11	6	5	1
5	6	11	17	9	7	16	4	1	5	6	11	17	9	7	16	4	1
6	17	7	4	5	11	9	16	1	6	17	7	4	5	11	9	16	1
7	11	1	7	11	1	7	11	1	7	11	1	7	11	1	7	11	1
8	7	18	11	12	1	8	7	18	11	12	1	8	7	18	11	12	1
9	5	7	6	16	11	4	17	1	9	5	7	6	16	11	4	17	1
10	5	12	6	3	11	15	17	18	9	14	7	13	16	8	4	2	1
11	7	1	11	7	1	11	7	1	11	7	1	11	7	1	11	7	1
12	11	18	7	8	1	12	11	18	7	8	1	12	11	18	7	8	1
13	17	12	4	14	11	10	16	18	6	2	7	15	5	8	9	3	1
14	6	8	17	10	7	3	4	18	5	13	11	2	9	12	16	15	1
15	16	12	9	2	11	13	5	18	4	3	7	10	17	8	6	14	1
16	9	11	5	4	7	17	6	1	16	9	11	5	4	7	17	6	1
17	4	11	16	6	7	5	9	1	17	4	11	16	6	7	5	9	1
18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1

Logarithms for Modular Arithmetic

With ordinary positive real numbers, the logarithm function is the inverse of exponentiation. An analogous function exists for modular arithmetic.

Let us briefly review the properties of ordinary logarithms. The logarithm of a number is defined to be the power to which some positive base (except 1) must be raised in order to equal the number. That is, for base x and for a value y,

$$y = x^{\log_x(y)}$$

The properties of logarithms include

$$\log_x(1) = 0$$
$$\log_x(x) = 1$$

$$\log_x(yz) = \log_x(y) + \log_x(z)$$

$$\log_x(y^r) = r \times \log_x(y)$$

Consider a primitive root a for some prime number p (the argument can be developed for nonprimes as well). Then we know that the powers of a from

1 through $(p - 1)$ produce each integer from 1 through $(p - 1)$ exactly once. We also know that any integer b satisfies

$$b \equiv r \pmod{p} \text{ for some } r, \text{ where } 0 \leq r \leq (p - 1)$$

by the definition of modular arithmetic. It follows that for any integer b and a primitive root a of prime number p, we can find a unique exponent i such that

$$b \equiv a^i \pmod{p} \quad \text{where } 0 \leq i \leq (p - 1)$$

This exponent i is referred to as the **discrete logarithm** of the number b for the base $a \pmod{p}$. We denote this value as $\text{dlog}_{a,p}(b)$.[11]

Note the following:

$$\text{dlog}_{a,p}(1) = 0 \text{ because } a^0 \bmod p = 1 \bmod p = 1$$

$$\text{dlog}_{a,p}(a) = 1 \text{ because } a^1 \bmod p = a$$

Here is an example using a nonprime modulus, $n = 9$. Here $\phi(n) = 6$ and $a = 2$ is a primitive root. We compute the various powers of a and find

$$\begin{array}{ll} 2^0 = 1 & 2^4 \equiv 7 \pmod{9} \\ 2^1 = 2 & 2^5 \equiv 5 \pmod{9} \\ 2^2 = 4 & 2^6 \equiv 1 \pmod{9} \\ 2^3 = 8 & \end{array}$$

This gives us the following table of the numbers with given discrete logarithms (mod 9) for the root $a = 2$:

Logarithm	0	1	2	3	4	5
Number	1	2	4	8	7	5

To make it easy to obtain the discrete logarithms of a given number, we rearrange the table:

Number	1	2	4	5	7	8
Logarithm	0	1	2	5	4	3

Now consider

$$x = a^{\text{dlog}_{a,p}(x)} \bmod p \qquad y = a^{\text{dlog}_{a,p}(y)} \bmod p$$
$$xy = a^{\text{dlog}_{a,p}(xy)} \bmod p$$

[11]Many texts refer to the discrete logarithm as the **index**. There is no generally agreed notation for this concept, much less an agreed name.

Using the rules of modular multiplication,

$$\begin{aligned} xy \bmod p &= [(x \bmod p)(y \bmod p)] \bmod p \\ a^{\mathrm{dlog}_{a,p}(xy)} \bmod p &= [(a^{\mathrm{dlog}_{a,p}(x)} \bmod p)(a^{\mathrm{dlog}_{a,p}(y)} \bmod p)] \bmod p \\ &= (a^{\mathrm{dlog}_{a,p}(x) + \mathrm{dlog}_{a,p}(y)}) \bmod p \end{aligned}$$

But now consider Euler's theorem, which states that, for every a and n that are relatively prime,

$$a^{\phi(n)} \equiv 1(\mathrm{mod}\ n)$$

Any positive integer z can be expressed in the form $z = q + k\phi(n)$, with $0 \leq q < \phi(n)$. Therefore, by Euler's theorem,

$$a^z \equiv a^q(\mathrm{mod}\ n) \qquad \text{if } z \equiv q \bmod \phi(n)$$

Applying this to the foregoing equality, we have

$$\mathrm{dlog}_{a,p}(xy) \equiv [\mathrm{dlog}_{a,p}(x) + \mathrm{dlog}_{a,p}(y)](\mathrm{mod}\ \phi(p))$$

and generalizing,

$$\mathrm{dlog}_{a,p}(y^r) \equiv [r \times \mathrm{dlog}_{a,p}(y)](\mathrm{mod}\ \phi(p))$$

This demonstrates the analogy between true logarithms and discrete logarithms.

Keep in mind that unique discrete logarithms mod m to some base a exist only if a is a primitive root of m.

Table 2.8, which is directly derived from Table 2.7, shows the sets of discrete logarithms that can be defined for modulus 19.

Calculation of Discrete Logarithms

Consider the equation

$$y = g^x \bmod p$$

Given g, x, and p, it is a straightforward matter to calculate y. At the worst, we must perform x repeated multiplications, and algorithms exist for achieving greater efficiency (see Chapter 9).

However, given y, g, and p, it is, in general, very difficult to calculate x (take the discrete logarithm). The difficulty seems to be on the same order of magnitude as that of factoring primes required for RSA. At the time of this writing, the asymptotically fastest known algorithm for taking discrete logarithms modulo a prime number is on the order of [BETH91]:

$$e^{((\ln p)^{1/3}(\ln(\ln p))^{2/3})}$$

which is not feasible for large primes.

Table 2.8 Tables of Discrete Logarithms, Modulo 19

(a) Discrete logarithms to the base 2, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$log_{2,19}(a)$	18	1	13	2	16	14	6	3	8	17	12	15	5	7	11	4	10	9

(b) Discrete logarithms to the base 3, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$log_{3,19}(a)$	18	7	1	14	4	8	6	3	2	11	12	15	17	13	5	10	16	9

(c) Discrete logarithms to the base 10, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$log_{10,19}(a)$	18	17	5	16	2	4	12	15	10	1	6	3	13	11	7	14	8	9

(d) Discrete logarithms to the base 13, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$log_{13,19}(a)$	18	11	17	4	14	10	12	15	16	7	6	3	1	5	13	8	2	9

(e) Discrete logarithms to the base 14, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$log_{14,19}(a)$	18	13	7	8	10	2	6	3	14	5	12	15	11	1	17	16	4	9

(f) Discrete logarithms to the base 15, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$log_{15,19}(a)$	18	5	11	10	8	16	12	15	4	13	6	3	7	17	1	2	14	9

2.9 KEY TERMS, REVIEW QUESTIONS, AND PROBLEMS

Key Terms

bijection	greatest common divisor	order
commutative	identity element	prime number
composite number	index	primitive root
discrete logarithm	modular arithmetic	relatively prime
divisor	modulus	residue

Review Questions

2.1 What does it mean to say that b is a divisor of a?

2.2 What is the meaning of the expression *a divides b*?

2.3 What is the difference between modular arithmetic and ordinary arithmetic?

2.4 What is a prime number?

2.5 What is Euler's totient function?

2.6 The Miller–Rabin test can determine if a number is not prime but cannot determine if a number is prime. How can such an algorithm be used to test for primality?

2.7 What is a primitive root of a number?

2.8 What is the difference between an index and a discrete logarithm?

Problems

2.1 Reformulate Equation (2.1), removing the restriction that a is a nonnegative integer. That is, let a be any integer.

2.2 Draw a figure similar to Figure 2.1 for $a < 0$.

2.3 For each of the following equations, find an integer x that satisfies the equation.
a. $4x \equiv 2 \pmod{3}$
b. $7x \equiv 4 \pmod{9}$
c. $5x \equiv 3 \pmod{11}$

2.4 In this text, we assume that the modulus is a positive integer. But the definition of the expression $a \bmod n$ also makes perfect sense if n is negative. Determine the following:
a. $7 \bmod 4$
b. $7 \bmod -4$
c. $-7 \bmod 4$
d. $-7 \bmod -4$

2.5 A modulus of 0 does not fit the definition but is defined by convention as follows: $a \bmod 0 = a$. With this definition in mind, what does the following expression mean: $a \equiv b \pmod{0}$?

2.6 In Section 2.3, we define the congruence relationship as follows: Two integers a and b are said to be congruent modulo n if $(a \bmod n) = (b \bmod n)$. We then proved that $a \equiv b \pmod{n}$ if $n|(a - b)$. Some texts on number theory use this latter relationship as the definition of congruence: Two integers a and b are said to be congruent modulo n if $n|(a - b)$. Using this latter definition as the starting point, prove that, if $(a \bmod n) = (b \bmod n)$, then n divides $(a - b)$.

2.7 What is the smallest positive integer that has exactly k divisors? Provide answers for values for $1 \leq k \leq 8$.

2.8 Prove the following:
a. $a \equiv b \pmod{n}$ implies $b \equiv a \pmod{n}$
b. $a \equiv b \pmod{n}$ and $b \equiv c \pmod{n}$ imply $a \equiv c \pmod{n}$

2.9 Prove the following:
a. $[(a \bmod n) - (b \bmod n)] \bmod n = (a - b) \bmod n$
b. $[(a \bmod n) \times (b \bmod n)] \bmod n = (a \times b) \bmod n$

2.10 Find the multiplicative inverse of each nonzero element in Z_5.

2.11 Show that an integer N is congruent modulo 9 to the sum of its decimal digits. For example, $723 \equiv 7 + 2 + 3 \equiv 12 \equiv 1 + 2 \equiv 3 \pmod{9}$. This is the basis for the familiar procedure of "casting out 9's" when checking computations in arithmetic.

2.12 **a.** Determine gcd(72345, 43215)
b. Determine gcd(3486, 10292)

2.13 The purpose of this problem is to set an upper bound on the number of iterations of the Euclidean algorithm.
a. Suppose that $m = qn + r$ with $q > 0$ and $0 \leq r < n$. Show that $m/2 > r$.
b. Let A_i be the value of A in the Euclidean algorithm after the ith iteration. Show that

$$A_{i+2} < \frac{A_i}{2}$$

c. Show that if m, n, and N are integers with $(1 \leq m, n, \leq 2^N)$, then the Euclidean algorithm takes at most $2N$ steps to find gcd(m, n).

2.14 The Euclidean algorithm has been known for over 2000 years and has always been a favorite among number theorists. After these many years, there is now a potential competitor, invented by J. Stein in 1961. Stein's algorithm is as follows: Determine gcd(A, B) with $A, B \geq 1$.
STEP 1 Set $A_1 = A, B_1 = B, C_1 = 1$
STEP 2 For $n > 1$, (1) If $A_n = B_n$, stop. gcd$(A, B) = A_nC_n$
(2) If A_n and B_n are both even, set $A_{n+1} = A_n/2$, $B_{n+1} = B_n/2$, $C_{n+1} = 2C_n$
(3) If A_n is even and B_n is odd, set $A_{n+1} = A_n/2$, $B_{n+1} = B_n$, $C_{n+1} = C_n$
(4) If A_n is odd and B_n is even, set $A_{n+1} = A_n$, $B_{n+1} = B_n/2$, $C_{n+1} = C_n$
(5) If A_n and B_n are both odd, set $A_{n+1} = |A_n - B_n|$, $B_{n+1} = \min(B_n, A_n)$, $C_{n+1} = C_n$
Continue to step $n + 1$.
a. To get a feel for the two algorithms, compute gcd(6150, 704) using both the Euclidean and Stein's algorithm.
b. What is the apparent advantage of Stein's algorithm over the Euclidean algorithm?

2.15 **a.** Show that if Stein's algorithm does not stop before the nth step, then

$$C_{n+1} \times \text{gcd}(A_{n+1}, B_{n+1}) = C_n \times \text{gcd}(A_n, B_n).$$

b. Show that if the algorithm does not stop before step $(n - 1)$, then

$$A_{n+2}B_{n+2} \leq \frac{A_nB_n}{2}.$$

c. Show that if $1 \leq A, B \leq 2^N$, then Stein's algorithm takes at most $4N$ steps to find gcd(m, n). Thus, Stein's algorithm works in roughly the same number of steps as the Euclidean algorithm.
d. Demonstrate that Stein's algorithm does indeed return gcd(A, B).

2.16 Using the extended Euclidean algorithm, find the multiplicative inverse of
a. 135 mod 61,
b. 7465 mod 2464, and
c. 42828 mod 6407.

2.17 The purpose of this problem is to determine how many prime numbers there are. Suppose there are a total of n prime numbers, and we list these in order: $p_1 = 2 < p_2 = 3 < p_3 = 5 < \ldots < p_n$.
a. Define $X = 1 + p_1p_2 \ldots p_n$. That is, X is equal to one plus the product of all the primes. Can we find a prime number P_m that divides X?
b. What can you say about m?
c. Deduce that the total number of primes cannot be finite.
d. Show that $P_{n+1} \leq 1 + p_1p_2 \ldots p_n$.

2.18 The purpose of this problem is to demonstrate that the probability that two random numbers are relatively prime is about 0.6.

a. Let $P = \Pr[\gcd(a, b) = 1]$. Show that $\Pr[\gcd(a, b) = d] = P/d^2$. *Hint:* Consider the quantity $\gcd\left(\frac{a}{d}, \frac{b}{d}\right)$.

b. The sum of the result of part (a) over all possible values of d is 1. That is $\Sigma^{d \geq 1}\Pr[\gcd(a, b) = d] = 1$. Use this equality to determine the value of P. *Hint:* Use the identity $\sum_{i=1}^{\infty}\frac{1}{i^2} = \frac{\pi^2}{6}$.

2.19 Why is $\gcd(n, n + 1) = 1$ for two consecutive integers n and $n + 1$?

2.20 Using Fermat's theorem, find 4^{225} mod 13.

2.21 Use Fermat's theorem to find a number a between 0 and 92 with a congruent to 7^{1013} modulo 93.

2.22 Use Fermat's theorem to find a number x between 0 and 37 with x^{73} congruent to 4 modulo 37. (You should not need to use any brute-force searching.)

2.23 Use Euler's theorem to find a number a between 0 and 9 such that a is congruent to 9^{101} modulo 10. (*Note*: This is the same as the last digit of the decimal expansion of 9^{100}.)

2.24 Use Euler's theorem to find a number x between 0 and 14 with x^{61} congruent to 7 modulo 15. (You should not need to use any brute-force searching.)

2.25 Notice in Table 2.6 that $\phi(n)$ is even for $n > 2$. This is true for all $n > 2$. Give a concise argument why this is so.

2.26 Prove the following: If p is prime, then $\phi(p^i) = p^i - p^{i-1}$. *Hint:* What numbers have a factor in common with p^i?

2.27 It can be shown (see any book on number theory) that if $\gcd(m, n) = 1$, then $\phi(mn) = \phi(m)\phi(n)$. Using this property, the property developed in the preceding problem, and the property that $\phi(p) = p - 1$ for p prime, it is straightforward to determine the value of $\phi(n)$ for any n. Determine the following:

a. $\phi(29)$ **b.** $\phi(51)$ **c.** $\phi(455)$ **d.** $\phi(616)$

2.28 It can also be shown that for arbitrary positive integer a, $\phi(a)$ is given by

$$\phi(a) = \prod_{i=1}^{t}[p_i^{a_i - 1}(p_i - 1)]$$

where a is given by Equation (2.9), namely: $a = P_1^{a_1}P_2^{a_2} \ldots P_t^{a_t}$. Demonstrate this result.

2.29 Consider the function: f(n) = number of elements in the set $\{a: 0 \leq a < n$ and $\gcd(a, n) = 1\}$. What is this function?

2.30 Although ancient Chinese mathematicians did good work coming up with their remainder theorem, they did not always get it right. They had a test for primality. The test said that n is prime if and only if n divides $(2^n - 2)$.

a. Give an example that satisfies the condition using an odd prime.

b. The condition is obviously true for $n = 2$. Prove that the condition is true if n is an odd prime (proving the **if** condition).

c. Give an example of an odd n that is not prime and that does not satisfy the condition. You can do this with nonprime numbers up to a very large value. This misled the Chinese mathematicians into thinking that if the condition is true then n is prime.

d. Unfortunately, the ancient Chinese never tried $n = 341$, which is nonprime ($341 = 11 \times 31$), yet 341 divides $2^{341} - 2$ without remainder. Demonstrate that $2341 \equiv 2 \pmod{341}$ (disproving the **only if** condition). *Hint:* It is not necessary to calculate 2^{341}; play around with the congruences instead.

2.31 Show that, if n is an odd composite integer, then the Miller–Rabin test will return `inconclusive` for $a = 1$ and $a = (n - 1)$.

2.32 If n is composite and passes the Miller–Rabin test for the base a, then n is called a *strong pseudoprime to the base a*. Show that 2047 is a strong pseudoprime to the base 2.

2.33 A common formulation of the Chinese remainder theorem (CRT) is as follows: Let $m_1, \ldots, m_k$ be integers that are pairwise relatively prime for $1 \leq i, j \leq k$, and $i \neq j$. Define M to be the product of all the m_i's. Let $a_1, \ldots, a_k$ be integers. Then the set of congruences:

$$\begin{aligned} x &\equiv a_1 (\text{mod } m_1) \\ x &\equiv a_2 (\text{mod } m_2) \\ &\vdots \\ x &\equiv a_k (\text{mod } m_k) \end{aligned}$$

has a unique solution modulo M. Show that the theorem stated in this form is true.

2.34 The example used by Sun-Tsu to illustrate the CRT was

$$x \equiv 2 \ (\text{mod } 3); x \equiv 3 \ (\text{mod } 5); x \equiv 2 \ (\text{mod } 7)$$

Solve for x.

2.35 Six professors begin courses on Monday, Tuesday, Wednesday, Thursday, Friday, and Saturday, respectively, and announce their intentions of lecturing at intervals of 3, 2, 5, 6, 1, and 4 days, respectively. The regulations of the university forbid Sunday lectures (so that a Sunday lecture must be omitted). When first will all six professors find themselves compelled to omit a lecture? *Hint*: Use the CRT.

2.36 Find all the primitive roots of 37.

2.37 Given 5 as a primitive root of 23, construct a table of discrete logarithms, and use it to solve the following congruences.

a. $3x^5 = 2 \ (\text{mod } 23)$
b. $7x^{10} + 1 = 0 \ (\text{mod } 23)$
c. $5^x = 6 \ (\text{mod } 23)$

Programming Problems

2.1 Write a computer program that implements fast exponentiation (successive squaring) modulo n.

2.2 Write a computer program that implements the Miller–Rabin algorithm for a user-specified n. The program should allow the user two choices: (1) specify a possible witness a to test using the Witness procedure or (2) specify a number s of random witnesses for the Miller–Rabin test to check.

APPENDIX 2A THE MEANING OF MOD

The operator mod is used in this book and in the literature in two different ways: as a binary operator and as a congruence relation. This appendix explains the distinction and precisely defines the notation used in this book regarding parentheses. This notation is common but, unfortunately, not universal.

The Binary Operator mod

If a is an integer and n is a positive integer, we define a mod n to be the remainder when a is divided by n. The integer n is called the **modulus**, and the remainder is called the **residue**. Thus, for any integer a, we can always write

$$a = \lfloor a/n \rfloor \times n + (a \bmod n)$$

Formally, we define the operator mod as

$$a \bmod n = a - \lfloor a/n \rfloor \times n \quad \text{for } n \neq 0$$

As a binary operation, mod takes two integer arguments and returns the remainder. For example, $7 \bmod 3 = 1$. The arguments may be integers, integer variables, or integer variable expressions. For example, all of the following are valid, with the obvious meanings:

$7 \bmod 3$

$7 \bmod m$

$x \bmod 3$

$x \bmod m$

$(x^2 + y + 1) \bmod (2m + n)$

where all of the variables are integers. For each of the above expressions, the value is the remainder that results when the left-hand term is divided by the right-hand term [see Equation (2.1)]. Note that if either the left- or right-hand argument is an expression, the expression is parenthesized. The operator mod is not inside parentheses.

In fact, the mod operation also works if the two arguments are arbitrary real numbers, not just integers. In this book, we are concerned only with the integer operation.

The Congruence Relation mod

As a congruence relation, mod expresses that two arguments have the same remainder with respect to a given modulus. For example, $7 \equiv 4 \pmod 3$ expresses the fact that both 7 and 4 have a remainder of 1 when divided by 3. The following two expressions are equivalent:

$$a \equiv b \pmod m \qquad \Leftrightarrow \qquad a \bmod m = b \bmod m$$

Another way of expressing it is to say that the expression $a \equiv b \pmod m$ is the same as saying that $a - b$ is an integral multiple of m. Again, all the arguments may be integers, integer variables, or integer variable expressions. For example, all of the following are valid, with the obvious meanings:

$7 \equiv 4 \pmod 3$

$x \equiv y \pmod m$

$(x^2 + y + 1) \equiv (a + 1)(\text{mod } [m + n])$

where all of the variables are integers. Two conventions are used. The congruence sign is $\equiv$. The modulus for the relation is defined by placing the mod operator followed by the modulus in parentheses.

The congruence relation is used to define **residue classes**. Those numbers that have the same remainder r when divided by m form a residue class (mod m). There are m residue classes (mod m). For a given remainder r, the residue class to which it belongs consists of the numbers

$$r, r \pm m, r \pm 2m, \ldots$$

According to our definition, the congruence

$$a \equiv b \pmod{m}$$

signifies that the numbers a and b differ by a multiple of m. Consequently, the congruence can also be expressed in the terms that a and b belong to the same residue class (mod m).

Chapter 3

Classical Encryption Techniques