**NETWORK SECURITY**

**ESSENTIALS:**

***APPLICATIONS AND STANDARDS* FOURTH EDITION**

**William Stallings**

**Prentice Hall**

Boston Columbus Indianapolis New York San Francisco Upper Saddle River Amsterdam Cape Town Dubai London Madrid Milan Munich Paris Montreal Toronto Delhi Mexico City Sao Paulo Sydney Hong Kong Seoul Singapore Taipei Tokyo

**Vice President and Editorial Director, ECS:** Marcia J. Horton

**Editor in Chief, Computer Science:** Michael Hirsch

**Executive Editor:** Tracy Dunkelberger **Assistant Editor:** Melinda Haggerty **Editorial Assistant:** Allison Michael

**Managing Editor:** Scott Disanno **Production Manager:** Wanda Rockwell **Art Director:** Jayne Conte

**Cover Designer:** Bruce Kenselaar **Cover Art:** Shutterstock

**Art Editor:** Greg Dulles

**Copyright © 2011 Pearson Education, Inc., publishing as [Prentice Hall, 1 Lake Street, Upper Saddle River, NJ 07458]**.All rights reserved. Manufactured in the United States of America.This publication is protected by Copyright, and permission should be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying,

recording, or likewise.To obtain permission(s) to use material from this work, please submit a written request to Pearson Education, Inc., Permissions Department, [imprint permissions address].

Many of the designations by manufacturers and seller to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed in initial caps or all caps.

**Library of Congress Cataloging-in-Publication Data**

10 9 8 7 6 5 4 3 2 1

ISBN 10: 0-13-610805-9

ISBN 13: 978-0-13-610805-4

*To Antigone never dull never boring always a Sage*

*This page intentionally left blank*

**CONTENTS**

**Preface ix**

**About the Author xiv**

**Chapter 1 Introduction 1**

**1.1** Computer Security Concepts 3

**1.2** The OSI Security Architecture 8

**1.3** Security Attacks 9

**1.4** Security Services 13

**1.5** Security Mechanisms 16

**1.6** A Model for Network Security 19

**1.7** Standards 21

**1.8** Outline of This Book 21

**1.9** Recommended Reading 22

**1.10** Internet and Web Resources 23

**1.11** Key Terms, Review Questions, and Problems 25

**PART ONE CRYPTOGRAPHY 27**

**Chapter 2 Symmetric Encryption and Message Confidentiality 27 2.1** Symmetric Encryption Principles 28

**2.2** Symmetric Block Encryption Algorithms 34

**2.3** Random and Pseudorandom Numbers 42

**2.4** Stream Ciphers and RC4 45

**2.5** Cipher Block Modes of Operation 50

**2.6** Recommended Reading and Web Sites 55

**2.7** Key Terms, Review Questions, and Problems 56

**Chapter 3 Public-Key Cryptography and Message Authentication 61 3.1** Approaches to Message Authentication 62

**3.2** Secure Hash Functions 67

**3.3** Message Authentication Codes 73

**3.4** Public-Key Cryptography Principles 79

**3.5** Public-Key Cryptography Algorithms 83

**3.6** Digital Signatures 90

**3.7** Recommended Reading and Web Sites 90

**3.8** Key Terms, Review Questions, and Problems 91

**PART TWO NETWORK SECURITY APPLICATIONS 97**

**Chapter 4 Key Distribution and User Authentication 97**

**4.1** Symmetric Key Distribution Using Symmetric Encryption 98

**4.2** Kerberos 99

**4.3** Key Distribution Using Asymmetric Encryption 114

**4.4** X.509 Certificates 116

**4.5** Public-Key Infrastructure 124**v**

**vi CONTENTS**

**4.6** Federated Identity Management 126

**4.7** Recommended Reading and Web Sites 132 **4.8** Key Terms, Review Questions, and Problems 133

**Chapter 5 Transport-Level Security 139**

**5.1** Web Security Considerations 140

**5.2** Secure Socket Layer and Transport Layer Security 143 **5.3** Transport Layer Security 156

**5.4** HTTPS 160

**5.5** Secure Shell (SSH) 162

**5.6** Recommended Reading and Web Sites 173 **5.7** Key Terms, Review Questions, and Problems 173

**Chapter 6 Wireless Network Security 175**

**6.1** IEEE 802.11 Wireless LAN Overview 177 **6.2** IEEE 802.11i Wireless LAN Security 183 **6.3** Wireless Application Protocol Overview 197 **6.4** Wireless Transport Layer Security 204

**6.5** WAP End-to-End Security 214

**6.6** Recommended Reading and Web Sites 217 **6.7** Key Terms, Review Questions, and Problems 218

**Chapter 7 Electronic Mail Security 221**

**7.1** Pretty Good Privacy 222

**7.2** S/MIME 241

**7.3** DomainKeys Identified Mail 257

**7.4** Recommended Reading and Web Sites 264 **7.5** Key Terms, Review Questions, and Problems 265 Appendix 7A Radix-64 Conversion 266

**Chapter 8 IP Security 269**

**8.1** IP Security Overview 270

**8.2** IP Security Policy 276

**8.3** Encapsulating Security Payload 281

**8.4** Combining Security Associations 288

**8.5** Internet Key Exchange 292

**8.6** Cryptographic Suites 301

**8.7** Recommended Reading and Web Sites 302 **8.8** Key Terms, Review Questions, and Problems 303

**PART THREE SYSTEM SECURITY 305**

**Chapter 9 Intruders 305**

**9.1** Intruders 307

**9.2** Intrusion Detection 312

**9.3** Password Management 323

**9.4** Recommended Reading and Web Sites 333 **9.5** Key Terms, Review Questions, and Problems 334 Appendix 9A The Base-Rate Fallacy 337

**CONTENTS vii**

**Chapter 10 Malicious Software 340**

**10.1** Types of Malicious Software 341

**10.2** Viruses 346

**10.3** Virus Countermeasures 351

**10.4** Worms 356

**10.5** Distributed Denial of Service Attacks 365

**10.6** Recommended Reading and Web Sites 370

**10.7** Key Terms, Review Questions, and Problems 371

**Chapter 11 Firewalls 374**

**11.1** The Need for Firewalls 375

**11.2** Firewall Characteristics 376

**11.3** Types of Firewalls 378

**11.4** Firewall Basing 385

**11.5** Firewall Location and Configurations 388

**11.6** Recommended Reading and Web Site 393

**11.7** Key Terms, Review Questions, and Problems 394

**APPENDICES 398**

**Appendix A Some Aspects of Number Theory 398**

**A.1** Prime and Relatively Prime Numbers 399

**A.2** Modular Arithmetic 401

**Appendix B Projects for Teaching Network Security 403**

**B.1** Research Projects 404

**B.2** Hacking Project 405

**B.3** Programming Projects 405

**B.4** Laboratory Exercises 406

**B.5** Practical Security Assessments 406

**B.6** Writing Assignments 406

**B.7** Reading/Report Assignments 407

**Index 408**

**ONLINE CHAPTERS**

**Chapter 12 Network Management Security**

**12.1** Basic Concepts of SNMP

**12.2** SNMPv1 Community Facility

**12.3** SNMPv3

**12.4** Recommended Reading and Web Sites

**12.5** Key Terms, Review Questions, and Problems

**Chapter 13 Legal and Ethical Aspects**

**13.1** Cybercrime and Computer Crime

**13.2** Intellectual Property

**13.3** Privacy

**13.4** Ethical Issues

**13.5** Recommended Reading and Web Sites

**viii CONTENTS**

**13.6** Key Terms, Review Questions, and Problems

**ONLINE APPENDICES**

**Appendix C Standards and Standards-Setting Organizations C.1** The Importance of Standards

**C.2** Internet Standards and the Internet Society

**C.3** National Institute of Standards and Technology

**Appendix D TCP/IP and OSI**

**D.1** Protocols and Protocol Architectures

**D.2** The TCP/IP Protocol Architecture

**D.3** The Role of an Internet Protocol

**D.4** IPv4

**D.5** IPv6

**D.6** The OSI Protocol Architecture

**Appendix E Pseudorandom Number Generation E.1** PRNG Requirements

**E.2** PRNG Using a Block Cipher

**E.3** PRNG Using a Hash Function or Message Authentication Code **Appendix F Kerberos Encryption Techniques**

**F.1** Password-to-Key Transformation

**F.2** Propagating Cipher Block Chaining Mode

**Appendix G Data Compression Using ZIP**

**G.1** Compression Algorithm

**G.2** Decompression Algorithm

**Appendix H PGP Random Number Generation**

**H.1** True Random Numbers

**H.2** Pseudorandom Numbers

**Appendix I The International Reference Alphabet Glossary**

**References**

**PREFACE**

*“The tie, if I might suggest it, sir, a shade more tightly knotted. One aims at the perfect butterfly effect. If you will permit me \_”*

*“What does it matter, Jeeves, at a time like this? Do you realize that Mr. Little’s domestic happiness is hanging in the scale?”*

*“There is no time, sir, at which ties do not matter.”*

—*Very Good, Jeeves!* P. G. Wodehouse

In this age of universal electronic connectivity, of viruses and hackers, of electronic eaves dropping and electronic fraud, there is indeed no time at which security does not matter.Two trends have come together to make the topic of this book of vital interest. First, the explosive growth in computer systems and their interconnections via networks has increased the dependence of both organizations and individuals on the information stored and communi cated using these systems. This, in turn, has led to a heightened awareness of the need to protect data and resources from disclosure, to guarantee the authenticity of data and messages, and to protect systems from network-based attacks. Second, the disciplines of cryptography and network security have matured, leading to the development of practical, readily available applications to enforce network security.

**OBJECTIVES**

It is the purpose of this book to provide a practical survey of network security applications and standards. The emphasis is on applications that are widely used on the Internet and for corpo rate networks, and on standards (especially Internet standards) that have been widely deployed.

**INTENDED AUDIENCE**

This book is intended for both an academic and a professional audience. As a textbook, it is intended as a one-semester undergraduate course on network security for computer science, computer engineering, and electrical engineering majors. It covers the material in IAS2 Security Mechanisms, a core area in the Information Technology body of knowledge; and NET4 Security, another core area in the Information Technology body of knowledge. These subject areas are part of the Draft ACM/IEEE Computer Society Computing Curricula 2005. The book also serves as a basic reference volume and is suitable for self-study.

**PLAN OF THE BOOK**

The book is organized in three parts:

**Part One. Cryptography:** A concise survey of the cryptographic algorithms and protocols underlying network security applications, including encryption, hash functions, digital signatures, and key exchange. **ix**

**x PREFACE**

**Part Two. Network Security Applications:** Covers important network security tools and applications, including Kerberos, X.509v3 certificates, PGP, S/MIME, IP Security, SSL/TLS, SET, and SNMPv3.

**Part Three. System Security:** Looks at system-level security issues, including the threat of and countermeasures for intruders and viruses and the use of firewalls and trusted systems. In addition, this book includes an extensive glossary, a list of frequently used acronyms, and a bibliography. Each chapter includes homework problems, review questions, a list of key words, suggestions for further reading, and recommended Web sites. In addition, a test bank is available to instructors.

**ONLINE DOCUMENTS FOR STUDENTS**

For this new edition, a tremendous amount of original supporting material has been made available online in the following categories.

• **Online chapters:** To limit the size and cost of the book, two chapters of the book are provided in PDF format. This includes a chapter on SNMP security and one on legal and ethical issues. The chapters are listed in this book’s table of contents.

• **Online appendices:** There are numerous interesting topics that support material found in the text but whose inclusion is not warranted in the printed text. Seven online appen dices cover these topics for the interested student.The appendices are listed in this book’s table of contents.

• **Homework problems and solutions:** To aid the student in understanding the material, a separate set of homework problems with solutions are provided. These enable the students to test their understanding of the text.

• **Supporting documents:** A variety of other useful documents are referenced in the text and provided online.

• **Key papers:** Twenty-Four papers from the professional literature, many hard to find, are provided for further reading.

Purchasing this textbook new grants the reader six months of access to this online material.

**INSTRUCTIONAL SUPPORT MATERIALS**

To support instructors, the following materials are provided.

• **Solutions Manual:** Solutions to end-of-chapter Review Questions and Problems. • **Projects Manual:** Suggested project assignments for all of the project categories listed subsequently in this Preface.

• **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.

All of these support materials are available at the Instructor Resource Center (IRC) for this textbook, which can be reached via pearsonhighered.com/stallings or by clicking on the button labeled “Book Info and More Instructor Resources” at this book’s Web site WilliamStallings.com/Crypto/Crypto5e.html. To gain access to the IRC, please contact your

**PREFACE xi**

local Prentice Hall sales representative via pearsonhighered.com/educator/replocator/ requestSalesRep.page or call Prentice Hall Faculty Services at 1-800-526-0485.

**INTERNET SERVICES FOR INSTRUCTORS AND STUDENTS**

There is a Web page for this book that provides support for students and instructors. The page includes links to other relevant sites, transparency masters of figures and tables in the book in PDF (Adobe Acrobat) format, and PowerPoint slides. The Web page is at **WilliamStallings.com/NetSec/NetSec4e.html**.

An Internet mailing list has been set up so that instructors using this book can exchange information, suggestions, and questions with each other and with the author. As soon as typos or other errors are discovered, an errata list for this book will be available at WilliamStallings.com. In addition, the Computer Science Student Resource site, at **WilliamStallings.com/StudentSupport.html**, provides documents, information, and useful links for computer science students and professionals.

**PROJECTS FOR TEACHING NETWORK SECURITY**

For many instructors, an important component of a 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 for including a projects compo nent in the course. The IRC not only includes guidance on how to assign and structure the projects, but also includes a set of suggested projects that covers a broad range of topics from the text:

• **Research projects:** A series of research assignments that instruct the student to research a particular topic on the Internet and write a report.

• **Hacking project:** This exercise is designed to illuminate the key issues in intrusion detection and prevention.

• **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. • **Lab exercises:** A series of projects that involve programming and experimenting with concepts from the book.

• **Practical security assessments:** A set of exercises to examine current infrastructure and practices of an existing organization.

• **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. See Appendix B for details.

**WHAT’S NEW IN THE FOURTH EDITION**

The changes for this new edition of *Network Security Essentials* are more substantial and comprehensive than those for any previous revision.

In the four years since the third edition of this book was published, the field has seen continued innovations and improvements. In this fourth edition, I try to capture these

**xii PREFACE**

changes while maintaining a broad and comprehensive coverage of the entire field. To begin this process of revision, the third edition was extensively reviewed by a number of professors who teach the subject. In addition, a number of professionals working in the field reviewed individual chapters. The result is that, in many places, the narrative has been clarified and tightened, and illustrations have been improved. Also, a large number of new “field-tested” problems have been added.

Beyond these refinements to improve pedagogy and user friendliness, there have been major substantive changes throughout the book. Highlights include:

• **Pseudorandom number generation and pseudorandom functions (revised):** The treat ment of this important topic has been expanded, with the addition of new material in Chapter 2 and a new appendix on the subject.

• **Cryptographic hash functions and message authentication codes (revised):** The material on hash functions and MAC has been revised and reorganized to provide a clearer and more systematic treatment.

• **Key distribution and remote user authentication (revised):** In the third edition, these topics were scattered across three chapters. In the fourth edition, the material is revised and consolidated into a single chapter to provide a unified, systematic treatment.

• **Federated identity (new):** A new section covers this common identity management scheme across multiple enterprises and numerous applications and supporting many thousands, even millions, of users.

• **HTTPS (new):** A new section covers this protocol for providing secure communication between Web browser and Web server.

• **Secure Shell (new):** SSH, one of the most pervasive applications of encryption tech nology, is covered in a new section.

• **DomainKeys Identified Mail (new):** A new section covers DKIM, which has become the standard means of authenticating e-mail to counter spam.

• **Wireless network security (new):** A new chapter covers this important area of network security. The chapter deals with the IEEE 802.11 (WiFi) security standard for wireless local area networks and the Wireless Application Protocol (WAP) security standard for communication between a mobile Web browser and a Web server.

• **IPsec (revised):** The chapter on IPsec has been almost completely rewritten. It now covers IPsecv3 and IKEv2. In addition, the presentation has been revised to improve clarity and breadth.

• **Legal and ethical issues (new):** A new online chapter covers these important topics. • **Online appendices (new):** Six online appendices provide addition breadth and depth for the interested student on a variety of topics.

• **Homework problems with solutions:** A separate set of homework problems (with solu tions) is provided online for students.

• **Test bank:** A test bank of review questions is available to instructors. This can be used for quizzes or to enable the students to check their understanding of the material. • **Firewalls (revised):** The chapter on firewalls has been significantly expanded. With each new edition, it is a struggle to maintain a reasonable page count while adding new material. In part, this objective is realized by eliminating obsolete material and tightening the narrative. For this edition, chapters and appendices that are of less general interest have

**PREFACE xiii**

been moved online as individual PDF files. This has allowed an expansion of material without the corresponding increase in size and price.

**RELATIONSHIP TO CRYPTOGRAPHY AND NETWORK SECURITY**

This book is adapted from *Cryptography and Network Security, Fifth Edition* (CNS5e). CNS5e provides a substantial treatment of cryptography, including detailed analysis of algorithms and a significant mathematical component, all of which covers 400 pages. *Network Security Essen tials:Applications and Standards, Fourth Edition* (NSE4e) provides instead a concise overview of these topics in Chapters 2 and 3. NSE4e includes all of the remaining material of CNS5e. NSE4e also covers SNMP security, which is not covered in CNS5e.Thus, NSE4e is intended for college courses and professional readers where the interest is primarily in the application of network security and without the need or desire to delve deeply into cryptographic theory and principles.

**ACKNOWLEDGEMENTS**

This new edition has benefited from review by a number of people who gave generously their time and expertise.The following people reviewed all or a large part of the manuscript: Marius Zimand (Towson State University), Shambhu Upadhyaya (University of Buffalo), Nan Zhang (George Washington University), Dongwan Shin (New Mexico Tech), Michael Kain (Drexel University), William Bard (University of Texas), David Arnold (Baylor University), Edward Allen (Wake Forest University), Michael Goodrich (UC-Irvine), Xunhua Wang (James Madison University), Xianyang Li (Illinois Institute of Technology), and Paul Jenkins (Brigham Young University).

Thanks also to the many people who provided detailed technical reviews of one or more chapters: Martin Bealby, Martin Hlavac (Department of Algebra, Charles University in Prague, Czech Republic), Martin Rublik (BSP Consulting and University of Economics in Bratislava), Rafael Lara (President of Venezuela’s Association for Information Security and Cryptography Research), Amitabh Saxena, and Michael Spratte (Hewlett-Packard Company). I would especially like to thank Nikhil Bhargava (IIT Delhi) for providing detailed reviews of various chapters of the book.

Nikhil Bhargava (IIT Delhi) developed the set of online homework problems and solutions. Professor Sreekanth Malladi of Dakota State University developed the hacking exercises. Sanjay Rao and Ruben Torres of Purdue 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 would like to thank the many people responsible for the publication of the book, all of whom did their usual excellent job. This includes my editor Tracy Dunkelberger and her assistants Melinda Hagerty and Allison Michael. Also, Jake Warde of Warde Publishers managed the reviews.

With all this assistance, little remains for which I can take full credit. However, I am proud to say that, with no help whatsoever, I selected all of the quotations.

**ABOUT THE AUTHOR**

William Stallings has made a unique contribution to understanding the broad sweep of tech nical developments in computer security, computer networking, and computer architecture. He has authored 17 titles and, counting revised editions, a total of 42 books on various aspects of these subjects. His writings have appeared in numerous ACM and IEEE publications, including the *Proceedings of the IEEE* and *ACM Computing Reviews*.

He has 11 times received the award 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. As a consultant, he has advised government agencies, computer and software vendors, and major users on the design, selection, and use of networking software and products.

He created and maintains the **Computer Science Student Resource Site** at WilliamStallings .com/StudentSupport.html. 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 M.I.T. in Computer Science and a B.S. from Notre Dame in electrical engineering.

**xiv**

**CHAPTER**

**INTRODUCTION**

**1.1 Computer Security Concepts**

A Definition of Computer Security

Examples

The Challenges of Computer Security

**1.2 The OSI Security Architecture**

**1.3 Security Attacks**

Passive Attacks

Active Attacks

**1.4 Security Services**

Authentication

Access Control

Data Confidentiality

Data Integrity

Nonrepudiation

Availability Service

**1.5 Security Mechanisms**

**1.6 A Model for Network Security**

**1.7 Standards**

**1.8 Outline of This Book**

**1.9 Recommended Reading**

**1.10 Internet and Web Resources**

Web Sites for This Book

Other Web Sites

USENET Newsgroups

**1.11 Key Terms, Review Questions, and Problems**

**1**

**2 CHAPTER 1 / INTRODUCTION**

*The combination of space, time, and strength that must be considered as the basic elements of this theory of defense makes this a fairly complicated matter. Conse quently, it is not easy to find a fixed point of departure.*

—*On War*, Carl Von Clausewitz

*The art of war teaches us to rely not on the likelihood of the enemy’s not coming, but on our own readiness to receive him; not on the chance of his not attacking, but rather on the fact that we have made our position unassailable.*

—*The Art of War,* Sun Tzu

The requirements of **information security** within an organization have undergone two major changes in the last several decades. Before the widespread use of data pro cessing equipment, the security of information felt to be valuable to an organization was provided primarily by physical and administrative means. An example of the former is the use of rugged filing cabinets with a combination lock for storing sensi tive documents.An example of the latter is personnel screening procedures used dur ing the hiring process.

With the introduction of the computer, the need for automated tools for protecting files and other information stored on the computer became evident. This is especially the case for a shared system, such as a time-sharing system, and the need is even more acute for systems that can be accessed over a public telephone network, data network, or the Internet. The generic name for the collection of tools designed to protect data and to thwart hackers is **computer security**.

The second major change that affected security is the introduction of distributed systems and the use of networks and communications facilities for carry ing data between terminal user and computer and between computer and computer. Network security measures are needed to protect data during their transmission. In fact, the term **network security** is somewhat misleading, because virtually all busi ness, government, and academic organizations interconnect their data processing equipment with a collection of interconnected networks. Such a collection is often referred to as an internet,1 and the term **internet security** is used.

There are no clear boundaries between these two forms of security. For exam ple, one of the most publicized types of attack on information systems is the com puter virus.A virus may be introduced into a system physically when it arrives on an optical disk and is subsequently loaded onto a computer. Viruses may also arrive over an internet. In either case, once the virus is resident on a computer system, internal computer security tools are needed to detect and recover from the virus.

This book focuses on internet security, which consists of measures to deter, prevent, detect, and correct security violations that involve the transmission of information. That is a broad statement that covers a host of possibilities. To give you a feel for the areas covered in this book, consider the following examples of security violations:

1We use the term *internet* with a lowercase “i” to refer to any interconnected collection of network. A corporate intranet is an example of an internet.The Internet with a capital “I” may be one of the facilities used by an organization to construct its internet.

**1.1 / COMPUTER SECURITY CONCEPTS 3**

**1.** User A transmits a file to user B. The file contains sensitive information (e.g., payroll records) that is to be protected from disclosure. User C, who is not authorized to read the file, is able to monitor the transmission and capture a copy of the file during its transmission.

**2.** A network manager, D, transmits a message to a computer, E, under its management.The message instructs computer E to update an authorization file to include the identities of a number of new users who are to be given access to that computer. User F intercepts the message, alters its contents to add or delete entries, and then forwards the message to E, which accepts the message as coming from manager D and updates its authorization file accordingly.

**3.** Rather than intercept a message, user F constructs its own message with the desired entries and transmits that message to E as if it had come from manager D. Computer E accepts the message as coming from manager D and updates its authorization file accordingly.

**4.** An employee is fired without warning. The personnel manager sends a message to a server system to invalidate the employee’s account.When the invalidation is accomplished, the server is to post a notice to the employee’s file as confirma tion of the action. The employee is able to intercept the message and delay it long enough to make a final access to the server to retrieve sensitive informa tion. The message is then forwarded, the action taken, and the confirmation posted.The employee’s action may go unnoticed for some considerable time.

**5.** A message is sent from a customer to a stockbroker with instructions for various transactions. Subsequently, the investments lose value and the customer denies sending the message.

Although this list by no means exhausts the possible types of security violations, it illus trates the range of concerns of network security.

This chapter provides a general overview of the subject matter that structures the material in the remainder of the book. We begin with a general discussion of network security services and mechanisms and of the types of attacks they are designed for. Then we develop a general overall model within which the security services and mechanisms can be viewed.

**1.1 COMPUTER SECURITY CONCEPTS**

**A Definition of Computer Security**

The NIST *Computer Security Handbook* [NIST95] defines the term *computer security* as **COMPUTER SECURITY**

The protection afforded to an automated information system in order to attain the applicable objectives of preserving the integrity, availability, and confidentiality of information system resources (includes hardware, software, firmware, information/ data, and telecommunications).

**4 CHAPTER 1 / INTRODUCTION**

This definition introduces three key objectives that are at the heart of com puter security.

• **Confidentiality:** This term covers two related concepts:

**Data**2 **confidentiality:** Assures that private or confidential information is not made available or disclosed to unauthorized individuals.

**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 information and programs are changed only in a specified and authorized manner.

**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** (Figure 1.1). The three concepts embody the fundamental security objectives for both data and for information and computing services. For example, the NIST *Standards for Security Categorization of Federal Information and Information Systems* (FIPS 199) lists confidentiality, integrity, and availability as the three security objectives for informa tion 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.

**lity**

**tia**

**n**

**e**

**fid**

**n**

**o**

**C**

**Data**

**and**

**services**

**Availability**

**Integrity**

**Figure 1.1** The Security Requirements

Triad

2RFC 2828 defines *information* as “facts and ideas, which can be represented (encoded) as various forms of data,” and *data* as “information in a specific physical representation, usually a sequence of symbols that have meaning; especially a representation of information that can be processed or produced by a computer.” Security literature typically does not make much of a distinction, nor does this book.

**1.1 / COMPUTER SECURITY CONCEPTS 5**

• **Confidentiality:** Preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and propri etary information. A loss of confidentiality is the unauthorized disclosure of information.

• **Integrity:** Guarding against improper information modification or des truction, 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 informa tion system.

Although the use of the CIA triad to define security objectives is well estab lished, some in the security field feel that additional concepts are needed to present a complete picture. Two of the most commonly mentioned are

• **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.

• **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 achiev able 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.

**Examples**

We now provide some examples of applications that illustrate the requirements just enumerated.3 For these examples, we use three levels of impact on organizations or individuals should there be a breach of security (i.e., a loss of confidentiality, integrity, or availability). These levels are defined in FIPS 199:

• **Low:** The loss could be expected to have a limited adverse effect on organiza tional operations, organizational assets, or individuals.A limited adverse effect means that, for example, the loss of confidentiality, integrity, or availability might (i) cause a degradation in mission capability to an extent and duration that the organization is able to perform its primary functions, but the effec tiveness of the functions is noticeably reduced; (ii) result in minor damage to organizational assets; (iii) result in minor financial loss; or (iv) result in minor harm to individuals.

3These examples are taken from a security policy document published by the Information Technology Security and Privacy Office at Purdue University.

**6 CHAPTER 1 / INTRODUCTION**

• **Moderate:** The loss could be expected to have a serious adverse effect on organizational operations, organizational assets, or individuals. A serious adverse effect means that, for example, the loss might (i) cause a significant degradation in mission capability to an extent and duration that the organi zation is able to perform its primary functions, but the effectiveness of the functions is significantly reduced; (ii) result in significant damage to organi zational assets; (iii) result in significant financial loss; or (iv) result in signifi cant harm to individuals that does not involve loss of life or serious, life-threatening injuries.

• **High:** The loss could be expected to have a severe or catastrophic adverse effect on organizational operations, organizational assets, or individuals. A severe or catastrophic adverse effect means that, for example, the loss might (i) cause a severe degradation in or loss of mission capability to an extent and duration that the organization is not able to perform one or more of its pri mary functions; (ii) result in major damage to organizational assets; (iii) result in major financial loss; or (iv) result in severe or catastrophic harm to individ uals involving loss of life or serious, life-threatening injuries.

***CONFIDENTIALITY*** Student grade information is an asset whose confidentiality is considered to be highly important by students. In the United States, the release of such information is regulated by the Family Educational Rights and Privacy Act (FERPA). Grade information should only be available to students, their parents, and employees that require the information to do their job. Student enrollment information may have a moderate confidentiality rating. While still covered by FERPA, this information is seen by more people on a daily basis, is less likely to be targeted than grade information, and results in less damage if disclosed. Directory information (such as lists of students, faculty, or departmental lists) may be assigned a low confidentiality rating or indeed no rating. This information is typically freely available to the public and published on a school’s Web site.

***INTEGRITY*** Several aspects of integrity are illustrated by the example of a hospital patient’s allergy information stored in a database. The doctor should be able to trust that the information is correct and current. Now suppose that an employee (e.g., a nurse) who is authorized to view and update this information deliberately falsifies the data to cause harm to the hospital. The database needs to be restored to a trusted basis quickly, and it should be possible to trace the error back to the person responsible. Patient allergy information is an example of an asset with a high requirement for integrity. Inaccurate information could result in serious harm or death to a patient and expose the hospital to massive liability.

An example of an asset that may be assigned a moderate level of integrity requirement is a Web site that offers a forum to registered users to discuss some specific topic. Either a registered user or a hacker could falsify some entries or deface the Web site. If the forum exists only for the enjoyment of the users, brings in little or no advertising revenue, and is not used for something important such as research, then potential damage is not severe. The Web master may experience some data, financial, and time loss.

**1.1 / COMPUTER SECURITY CONCEPTS 7**

An example of a low-integrity requirement is an anonymous online poll. Many Web sites, such as news organizations, offer these polls to their users with very few safeguards. However, the inaccuracy and unscientific nature of such polls is well understood.

***AVAILABILITY*** The more critical a component or service, the higher is the level of availability required. Consider a system that provides authentication services for critical systems, applications, and devices. An interruption of service results in the inability for customers to access computing resources and for the staff to access the resources they need to perform critical tasks. The loss of the service translates into a large financial loss due to lost employee productivity and potential customer loss.

An example of an asset that typically would be rated as having a moderate availability requirement is a public Web site for a university; the Web site provides information for current and prospective students and donors. Such a site is not a critical component of the university’s information system, but its unavailability will cause some embarrassment.

An online telephone directory lookup application would be classified as a low availability requirement. Although the temporary loss of the application may be an annoyance, there are other ways to access the information, such as a hardcopy direc tory or the operator.

**The Challenges of Computer Security**

Computer and network security is both fascinating and complex. Some of the reasons include:

**1.** Security is not as simple as it might first appear to the novice. The require ments seem to be straightforward; indeed, most of the major requirements for security services can be given self-explanatory, one-word labels: confidential ity, authentication, nonrepudiation, 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 elab orate 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].

**8 CHAPTER 1 / INTRODUCTION**

**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 cre ation, 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 net work that introduces variable, unpredictable delays may render such time limits meaningless.

**6.** Computer and network security is essentially a battle of wits between a perpetra tor 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 com puter and network 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-T4 Recommendation X.800, *Security Architecture for OSI*, defines such a systematic approach.5 The OSI security architecture is useful to managers as a way

4The International Telecommunication Union (ITU) Telecommunication Standardization Sector (ITU-T) is a United Nations-sponsored agency that develops standards, called Recommendations, relating to telecommunications and to open systems interconnection (OSI).

5The OSI security architecture was developed in the context of the OSI protocol architecture, which is described in Appendix D. However, for our purposes in this chapter, an understanding of the OSI protocol architecture is not required.

**Table 1.1** Threats and Attacks (RFC 2828) **Threat**

**1.3 / SECURITY ATTACKS 9**

A potential for violation of security, which exists when there is a circumstance, capability, action, or event that could breach security and cause harm. That is, a threat is a possible danger that might exploit a vulnerability.

**Attack**

An assault on system security that derives from an intelligent threat. That is, an intelligent act that is a deliberate attempt (especially in the sense of a method or technique) to evade security services and violate the security policy of a system.

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 archi tecture 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 to mean more or less the same thing. Table 1.1 provides definitions taken from RFC 2828, *Internet Security Glossary*.

**1.3 SECURITY ATTACKS**

A useful means of classifying security attacks, used both in X.800 and RFC 2828, is in terms of *passive attacks* and *active attacks*. 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 opponent is to obtain information that is being transmitted.Two types of passive attacks are the release of message contents and traffic analysis.

**10 CHAPTER 1 / INTRODUCTION Darth**

Read contents of message from Bob to Alice

**Bob**

**Internet or**

**other comms facility**

**(a) Release of message contents**

**Alice**

**Darth**

Observe pattern of messages from Bob to Alice

**Bob**

**Internet or**

**other comms facility (b) Traffic analysis**

**Alice**

**Figure 1.2** Passive Network Security Attacks

The **release of message contents** is easily understood (Figure 1.2a). A tele phone 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.

**1.3 / SECURITY ATTACKS 11**

A second type of passive attack,**traffic analysis**, is subtler (Figure 1.2b). 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 still might 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 communica tion 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 the 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: masquerade, replay, modification of messages, and denial of service.

A **masquerade** takes place when one entity pretends to be a different entity (Figure 1.3a). A masquerade attack usually includes one of the other forms of active attack. For example, authentication sequences can be cap tured and replayed after a valid authentication sequence has taken place, thus enabling an authorized entity with few privileges to obtain extra privi leges 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 (Figure 1.3b). **Modification of messages** simply means that some portion of a legitimate message is altered, or that messages are delayed or reordered, to produce an unauthorized effect (Figure 1.3c). For example, a message meaning “Allow John Smith to read confidential file accounts” is modified to mean “Allow Fred Brown to read confidential file accounts.”

The **denial of service** prevents or inhibits the normal use or manage ment of communications facilities (Figure 1.3d). 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 dis abling 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

**12 CHAPTER 1 / INTRODUCTION**

prevent their success. On the other hand, it is quite difficult to prevent active attacks absolutely because of the wide variety of potential physical, software, and network vulnerabilities. Instead, the goal is to detect active attacks and to recover from any dis ruption or delays caused by them. If the detection has a deterrent effect, it also may contribute to prevention.

**Darth**

Message from Darth that appears to be from Bob

**Bob**

**Bob**

**Figure 1.3** Active Attacks

**Internet or**

**other comms facility**

**(a) Masquerade**

**Darth** Capture message from Bob to Alice; later

replay message to Alice

**Internet or**

**other comms facility**

**(b) Replay**

**Alice Alice**

**Darth**

Darth modifies

**1.4 / SECURITY SERVICES 13**

**Bob**

message from Bob

to Alice

**Internet or**

**other comms facility**

**(c) Modification of messages**

**Alice**

**Darth**

Darth disrupts service provided by server

**Bob**

**Internet or**

**other comms facility (d) Denial of service**

**Server**

**Figure 1.3** Active Attacks (*Continued*)

**1.4 SECURITY SERVICES**

X.800 defines a security service as a service that is provided by a protocol layer of communicating open systems and that ensures adequate security of the systems or of data transfers. Perhaps a clearer definition is found in RFC 2828, which provides the following definition: A processing or communication service that is provided by

**14 CHAPTER 1 / INTRODUCTION**

a system to give a specific kind of protection to system resources; security services implement security policies and are implemented by security mechanisms. X.800 divides these services into five categories and fourteen specific services (Table 1.2). We look at each category in turn.6

**Table 1.2** Security Services (X.800)

**AUTHENTICATION**

The assurance that the communicating entity is the one that it claims to be.

**Peer Entity Authentication**

Used in association with a logical connection to provide confidence in the identity of the entities connected.

**Data-Origin Authentication**

In a connectionless transfer, provides assurance that the source of received data is as claimed.

**ACCESS CONTROL**

The prevention of unauthorized use of a resource (i.e., this service controls who can have access to a resource, under what conditions access can occur, and what those accessing the resource are allowed to do).

**DATA CONFIDENTIALITY**

The protection of data from unauthorized disclosure.

**Connection Confidentiality**

The protection of all user data on a connection.

**Connectionless Confidentiality**

The protection of all user data in a single data block.

**Selective-Field Confidentiality**

The confidentiality of selected fields within the user data on a connection or in a single data block.

**Traffic-Flow Confidentiality**

The protection of the information that might be derived from observation of traffic flows.

**DATA INTEGRITY**

The assurance that data received are exactly as sent by an authorized entity (i.e., contain no modification, insertion, deletion, or replay).

**Connection Integrity with Recovery**

Provides for the integrity of all user data on a connection and detects any modification, insertion, deletion, or replay of any data within an entire data sequence, with recovery attempted.

**Connection Integrity without Recovery**

As above, but provides only detection without recovery.

**Selective-Field Connection Integrity**

Provides for the integrity of selected fields within the user data of a data block transferred over a connec tion and takes the form of determination of whether the selected fields have been modified, inserted, deleted, or replayed.

**Connectionless Integrity**

Provides for the integrity of a single connectionless data block and may take the form of detection of data modification. Additionally, a limited form of replay detection may be provided.

**Selective-Field Connectionless Integrity** Provides for the integrity of selected fields within a single connectionless data block; takes the form of determina tion of whether the selected fields have been modified.

**NONREPUDIATION**

Provides protection against denial by one of the entities involved in a communication of having participated in all or part of the communication.

**Nonrepudiation, Origin**

Proof that the message was sent by the specified party.

**Nonrepudiation, Destination**

Proof that the message was received by the specified party.

6There 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 both X.800 and RFC 2828.

**1.4 / SECURITY SERVICES 15**

**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 terminal to a host, 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 imple ment the same protocol in different systems (e.g., two TCP modules in two communicating systems). Peer entity authentication is provided for use at the establishment of or 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 prior 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 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 des tination, frequency, length, or other characteristics of the traffic on a communica tions facility.

**16 CHAPTER 1 / INTRODUCTION**

**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 deals with a stream of messages and assures that messages are received as sent with no duplication, insertion, mod ification, 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 deals with individual messages without regard to any larger con text and 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 typically the more attractive alternative.

**Nonrepudiation**

**Nonrepudiation** prevents either sender or receiver from denying a transmitted mes sage. 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**

Both X.800 and RFC 2828 define **availability** to be the property of a system or a sys tem resource being accessible and usable upon demand by an authorized system entity, according to performance specifications for the system (i.e., a system is avail able 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 authentica tion 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**

Table 1.3 lists the security mechanisms defined in X.800.The mechanisms are divided into those that are implemented in a specific protocol layer, such as TCP or an application layer protocol, and those that are not specific to any particular protocol layer or security

**1.5 / SECURITY MECHANISMS 17**

**Table 1.3** Security Mechanisms (X.800)

**SPECIFIC SECURITY MECHANISMS**

May be incorporated into the appropriate protocol layer in order to provide some of the OSI security services.

**Encipherment**

The use of mathematical algorithms to transform data into a form that is not readily intelligible. The transformation and subsequent recovery of the data depend on an algorithm and zero or more encryption keys.

**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 (e.g., by the recipient).

**Access Control**

A variety of mechanisms that enforce access rights to resources.

**Data Integrity**

A variety of mechanisms used to assure the integrity of a data unit or stream of data units.

**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 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.

**PERVASIVE SECURITY MECHANISMS**

Mechanisms that are not specific to any particular OSI security service or protocol layer.

**Trusted Functionality**

That which is perceived to be correct with respect to some criteria (e.g., as established by a security policy).

**Security Label**

The marking bound to a resource (which may be a data unit) that names or designates the security attributes of that resource.

**Event Detection**

Detection of security-relevant events.

**Security Audit Trail**

Data collected and potentially used to facilitate a security audit, which is an independent review and examination of system records and activities.

**Security Recovery**

Deals with requests from mechanisms, such as event handling and management functions, and takes recovery actions.

service.These mechanisms will be covered in the appropriate places in the book, so we do not elaborate now except to comment on the definition of encipherment. X.800 distin guishes between reversible encipherment mechanisms and irreversible encipherment mechanisms. A reversible encipherment mechanism is simply an encryption algorithm that allows data to be encrypted and subsequently decrypted. Irreversible encipherment mechanisms include hash algorithms and message authentication codes,which are used in digital signature and message authentication applications.

Table 1.4, based on one in X.800, indicates the relationship between security services and security mechanisms.

**18**

s

m

s

i

n

a

hc

e

M

d

n

a

s

e

c

i

v

r

e

S

y

t

ir

u

c

e

S

n

e

e

w

t

e

B

pi

hs

n

o

it

a

le

R

**4 .**

**1**

**e**

**l**

**b**

**a**

**T**

**m**

**s**

**i**

**n**

**a**

**hc**

**e**

**M**

**a**

**r**

**T**

**n**

**o**

**i**

**t**

**a**

**c**

**i**

**t**

**n**

**e**

**h**

**t**

**u**

**A**

**a**

**t**

**a**

**D**

**s**

**s**

**e**

**c**

**c**

**A**

**l**

**a**

**t**

**i**

**gi**

**D**

**d**

**a**

**P**

**e**

**g**

**n**

**a**

**h**

**c**

**x**

**E**

**y**

**t**

**i**

**r**

**g**

**e**

**t**

**n**

**I**

**l**

**o**

**r**

**t**

**n**

**o**

**C**

**e**

**r**

**u**

**t**

**a**

**n**

**gi**

**S**

**t**

**n**

**e**

**m**

**r**

**e**

**h**

**p**

**i**

**c**

**n**

**E**

**e**

**c**

**i**

**v**

**r**

**e**

**S**

Y

Y

Y

n

o

i

t

a

c

i

t

n

e

h

t

u

A

y

t

i

t

n

E

r

e

e

P

Y

Y

n

o

it

a

c

it

n

e

ht

u

A

n

i

gi

r

O-

a

t

a

D

Y

l

o

r

tn

o

C

s

s

e

c

c

A

Y

y

t

il

a

it

n

e

di

f

n

o

C

Y

y

t

il

a

it

n

e

di

f

n

o

C

w

o

l

F

-

c

if

f

a

r

T

Y

Y Y

y

t

ir

g

e

t

n

I

a

t

a

D

Y Y

n

o

it

a

i

du

p

e

r

n

o

N

Y Y

y

t

il

i

b

a

li

a

v

A

**1.6 / A MODEL FOR NETWORK SECURITY 19**

**1.6 A MODEL FOR NETWORK SECURITY**

A model for much of what we will be discussing is captured, in very general terms, in Figure 1.4. A message is to be transferred from one party to another across some sort of Internet service. The two parties, who are the *principals* in this transaction, must cooperate for the exchange to take place. A logical information channel is established by defining a route through the Internet from source to destination and by the cooperative use of communication protocols (e.g., TCP/IP) by the two principals.

Security aspects come into play when it is necessary or desirable to protect the information transmission from an opponent who may present a threat to confidentiality, authenticity, and so on. All of the techniques for providing security have two components:

**1.** A security-related transformation on the information to be sent. Examples include the encryption of the message, which scrambles the message so that it is unreadable by the opponent, and the addition of a code based on the contents of the message, which can be used to verify the identity of the sender.

**2.** Some secret information shared by the two principals and, it is hoped, unknown to the opponent.An example is an encryption key used in conjunction with the transformation to scramble the message before transmission and unscramble it on reception.7

**Trusted third party**

**(e.g., arbiter, distributer**

**of secret information)**

**Sender**

**e**

**g**

**a**

**s**

**s**

**e**

**M**

**Information**

**channel Security-related transformation**

**e**

**e**

**e**

**e**

**g**

**g**

**r**

**r**

**a**

**a**

**u**

**u**

**s**

**s**

**c**

**c**

**s**

**s**

**e**

**e**

**e**

**e**

**S**

**S**

**m**

**m**

**Secret**

**information**

**Opponent**

**Recipient**

**Security-related**

**transformation**

**e**

**g**

**a**

**s**

**s**

**e**

**M**

**Secret**

**information**

**Figure 1.4** Model for Network Security

7Chapter 3 discusses a form of encryption, known as asymmetric encryption, in which only one of the two principals needs to have the secret information.

**20 CHAPTER 1 / INTRODUCTION**

A trusted third party may be needed to achieve secure transmission. For example, a third party may be responsible for distributing the secret information to the two principals while keeping it from any opponent. Or a third party may be needed to arbitrate disputes between the two principals concerning the authenticity of a message transmission.

This general model shows that there are four basic tasks in designing a particular security service:

**1.** Design an algorithm for performing the security-related transformation. The algorithm should be such that an opponent cannot defeat its purpose. **2.** Generate the secret information to be used with the algorithm.

**3.** Develop methods for the distribution and sharing of the secret information. **4.** Specify a protocol to be used by the two principals that makes use of the security algorithm and the secret information to achieve a particular security service.

Parts One and Two of this book concentrate on the types of security mechanisms and services that fit into the model shown in Figure 1.4. However, there are other security-related situations of interest that do not neatly fit this model but are consid ered in this book.A general model of these other situations is illustrated by Figure 1.5, which reflects a concern for protecting an information system from unwanted access. Most readers are familiar with the concerns caused by the existence of hackers who attempt to penetrate systems that can be accessed over a network. The hacker can be someone who, with no malign intent, simply gets satisfaction from breaking and enter ing a computer system. The intruder can be a disgruntled employee who wishes to do damage or a criminal who seeks to exploit computer assets for financial gain (e.g., obtaining credit card numbers or performing illegal money transfers).

Another type of unwanted access is the placement in a computer system of logic that exploits vulnerabilities in the system and that can affect application pro grams as well as utility programs, such as editors and compilers. Programs can pre sent two kinds of threats:

**1. Information access threats:** Intercept or modify data on behalf of users who should not have access to that data.

**2. Service threats:** Exploit service flaws in computers to inhibit use by legitimate users.

**Information System**

**Computing resources**

**Opponent**

**— human (e.g., hacker)**

**— software**

**(e.g., virus, worm)**

**Access Channel**

**Figure 1.5** Network Access Security Model

**Gatekeeper function**

**(processor, memory, I/O) Data**

**Processes**

**Software**

**Internal security controls**

**1.8 / OUTLINE OF THIS BOOK 21**

Viruses and worms are two examples of software attacks. Such attacks can be introduced into a system by means of a disk that contains the unwanted logic con cealed in otherwise useful software. They also can be inserted into a system across a network; this latter mechanism is of more concern in network security.

The **security mechanisms** needed to cope with unwanted access fall into two broad categories (see Figure 1.5). The first category might be termed a gatekeeper function. It includes password-based login procedures that are designed to deny access to all but authorized users and screening logic that is designed to detect and reject worms, viruses, and other similar attacks. Once either an unwanted user or unwanted software gains access, the second line of defense consists of a variety of internal controls that monitor activity and analyze stored information in an attempt to detect the presence of unwanted intruders.These issues are explored in Part Three.

**1.7 STANDARDS**

Many of the security techniques and applications described in this book have been specified as standards. Additionally, standards have been developed to cover man agement 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 organi zations 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 address ing 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 specifica tions, all of which are published as **Requests for Comments (RFCs)**.

A more detailed discussion of these organizations is contained in Appendix C. **1.8 OUTLINE OF THIS BOOK**

This chapter serves as an introduction to the entire book.The remainder of the book is organized into three parts.

**Part One:** Provides a concise survey of the cryptographic algorithms and proto cols underlying network security applications, including encryption,

hash functions, and digital signatures.

**22 CHAPTER 1 / INTRODUCTION**

**Part Two:** Examines the use of cryptographic algorithms and security proto cols to provide security over networks and the Internet.Topics cov

ered include key management, user authentication, transport-level

security, wireless network security, e-mail security, and IP security.

**Part Three:** Deals with security facilities designed to protect a computer sys tem from security threats, including intruders, viruses, and worms.

This part also looks at firewall technology.

In addition, two online chapters cover network management security and legal and ethical issues.

**1.9 RECOMMENDED READING**

[STAL08] provides a broad introduction to computer security. [SCHN00] is valuable reading for any practitioner in the field of computer or network security: It discusses the limitations of technology (and cryptography in particular) in providing security and the need to consider the hardware, the software implementation, the networks, and the people involved in provid ing and attacking security.

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 papers to read are [WARE79], [BROW72], [SALT75], [SHAN77], and [SUMM84]. Two more recent, short treatments of computer security are [ANDR04] and [LAMP04]. [NIST95] is an exhaustive (290 pages) treatment of the subject. Another good treatment is [NRC91]. Also useful is [FRAS97].

**ANDR04** Andrews, M., and Whittaker, J. “Computer Security.” *IEEE Security and Privacy*, September/October 2004.

**BROW72** Browne, P. “Computer Security — A Survey.”*ACM SIGMIS Database*, Fall 1972. **FRAS97** Fraser, B. *Site Security Handbook.* RFC 2196, September 1997.

**LAMP04** Lampson, B. “Computer Security in the Real World.” *Computer*, June 2004. **NIST95** National Institute of Standards and Technology. *An Introduction to Computer Security: The NIST Handbook.* Special Publication 800–12. October 1995.

**NRC91** National Research Council. *Computers at Risk: Safe Computing in the Information Age.* Washington, D.C.: National Academy Press, 1991.

**SALT75** Saltzer, J., and Schroeder, M. “The Protection of Information in Computer Systems.” *Proceedings of the IEEE*, September 1975.

**SCHN00** Schneier, B. *Secrets and Lies: Digital Security in a Networked World.* New York: Wiley 2000.

**SHAN77** Shanker, K. “The Total Computer Security Problem: An Overview.” *Computer*, June 1977.

**STAL08** Stallings, W., and Brown, L. *Computer Security*. Upper Saddle River, NJ: Prentice Hall, 2008.

**SUMM84** Summers, R. “An Overview of Computer Security.” *IBM Systems Journal* , Vol. 23, No. 4, 1984.

**WARE79** Ware, W., ed. *Security Controls for Computer Systems.* RAND Report 609–1. October 1979. http://www.rand.org/pubs/reports/R609-1/R609.1.html

**1.10 / INTERNET AND WEB RESOURCES 23**

**1.10 INTERNET AND WEB RESOURCES**

There are a number of resources available on the Internet and the Web to support this book and to help one keep up with developments in this field.

**Web Sites for This Book**

There is a Web page for this book at **WilliamStallings.com/NetSec/NetSec4e.html**. The site includes the following:

• **Useful Web sites:** There are links to other relevant Web sites organized by chapter, including the sites listed in this section and throughout this book.

• **Online documents:** Link to the Companion Website at Pearson that includes supple mental online chapters and appendices, homework problems and solutions, impor tant papers from the literature, and other supporting documents. See Preface for details.

• **Errata sheet:** An errata list for this book will be maintained and updated as needed. Please e-mail any errors that you spot to me. Errata sheets for my other books are at **WilliamStallings.com**.

• **Internet mailing list:** The site includes sign-up information for the book’s Internet mail ing list.

• **Network security courses:** There are links to home pages for courses based on this book; these pages may be useful to instructors in providing ideas about how to struc ture their course.

I also maintain the Computer Science Student Resource Site at **WilliamStallings.com/ StudentSupport.html**.The purpose of this site is to provide documents, information, and links for computer science students and professionals. Links and documents are organized into six categories:

• **Math:** Includes a basic math refresher, a queuing analysis primer, a number system primer, and links to numerous math sites.

• **How-to:** Advice and guidance for solving homework problems, writing technical reports, and preparing technical presentations.

• **Research resources:** Links to important collections of papers, technical reports, and bibliographies.

• **Computer science careers:** Useful links and documents for those considering a career in computer science.

• **Miscellaneous:** A variety of other interesting documents and links.

• **Humor and other diversions:** You have to take your mind off your work once in a while.

**24 CHAPTER 1 / INTRODUCTION**

**Other Web Sites**

There are numerous Web sites that provide information related to the topics of this book. In subsequent chapters, pointers to specific Web sites can be found in the *Recommended Reading and Web Sites* section. Because the addresses for Web sites tend to change frequently, I have not included URLs in the book. For all of the Web sites listed in the book, the appropriate link can be found at this book’s Web site. Other links not mentioned in this book will be added to the Web site over time.

The following Web sites are of general interest related to cryptography and network security.

• **IETF Security Area:** Material related to Internet security standardization efforts. • **Computer and Network Security Reference Index:** A good index to vendor and commercial products, frequently asked questions (FAQs), newsgroup archives, papers, and other Web sites.

• **The Cryptography FAQ:** Lengthy and worthwhile FAQ covering all aspects of cryptography.

• **Tom Dunigan’s Security Page:** An excellent list of pointers to cryptography and network security Web sites.

• **Helger Lipmaa’s Cryptology Pointers:** Another excellent list of pointers to cryptography and network security Web sites.

• **IEEE Technical Committee on Security and Privacy:** Copies of their newsletter and information on IEEE-related activities.

• **Computer Security Resource Center:** Maintained by the National Institute of Standards and Technology (NIST); contains a broad range of information on security threats, technology, and standards.

• **Security Focus:** A wide variety of security information with an emphasis on vendor products and end-user concerns.

• **SANS Institute:** Similar to Security Focus. Extensive collection of white papers. • **Center for Internet Security:** Provides freeware benchmark and scoring tools for eval uating security of operating systems, network devices, and applications. Includes case studies and technical papers.

• **Institute for Security and Open Methodologies:** An open, collaborative security research community. Lots of interesting information.

**USENET Newsgroups**

A number of USENET newsgroups are devoted to some aspect of network security or cryp tography. As with virtually all USENET groups, there is a high noise-to-signal ratio, but it is worth experimenting to see if any meet your needs. The most relevant are the following:

• **sci.crypt.research:** The best group to follow. This is a moderated newsgroup that deals with research topics; postings must have some relationship to the technical aspects of cryptology.

**1.11 / KEY TERMS, REVIEW QUESTIONS,AND PROBLEMS 25**

• **sci.crypt:** A general discussion of cryptology and related topics.

• **sci.crypt.random-numbers:** A discussion of cryptographic strength randomness. • **alt.security:** A general discussion of security topics.

• **comp.security.misc:** A general discussion of computer security topics.

• **comp.security.firewalls:** A discussion of firewall products and technology.

• **comp.security.announce:** News and announcements from CERT.

• **comp.risks:** A discussion of risks to the public from computers and users.

• **comp.virus:** A moderated discussion of computer viruses.

In addition, there are a number of forums dealing with cryptography available on the Internet. Among the most worthwhile are

• **Security and Cryptography forum:** Sponsored by DevShed. Discusses issues related to cod ing, server applications, network protection, data protection, firewalls, ciphers, and the like. • **Cryptography forum:** On Topix. Fairly good focus on technical issues.

• **Security forums:** On WindowsSecurity.com. Broad range of forums, including crypto graphic theory, cryptographic software, firewalls, and malware.

Links to these forums are provided at this book’s Web site.

**1.11 KEY TERMS, REVIEW QUESTIONS,AND PROBLEMS Key Terms**

access control

active threat

authentication

authenticity

availability

data confidentiality

data integrity

**Review Questions**

denial of service

encryption

integrity

intruder

masquerade

nonrepudiation

OSI security architecture

passive threat

replay

security attacks

security mechanisms security services traffic analysis

**1.1** What is the OSI security architecture?

**1.2** What is the difference between passive and active security threats? **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.

**Problems**

**1.1** Consider an automated teller machine (ATM) in which users provide a personal identification number (PIN) and a card for account access. Give examples of confi dentiality, integrity, and availability requirements associated with the system. In each case, indicate the degree of importance of the requirement.

**26 CHAPTER 1 / INTRODUCTION**

**1.2** Repeat Problem 1.1 for a telephone switching system that routes calls through a switching network based on the telephone number requested by the caller.

**1.3** Consider a desktop publishing system used to produce documents 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.** An organization managing public information on its Web server.

**b.** A law-enforcement organization managing extremely sensitive investigative information.

**c.** A financial organization managing routine administrative information (not privacy-related information).

**d.** An information system used for large acquisitions in a contracting organization that contains both sensitive, pre-solicitation phase contract information and routine administrative information. Assess the impact for the two data sets separately and the information system as a whole.

**e.** A power plant contains a SCADA (supervisory control and data acquisition) system controlling the distribution of electric power for a large military installation. The SCADA system contains both real-time sensor data and routine administrative information. Assess the impact for the two data sets separately and the information system as a whole.

**1.5** Draw a matrix similar to Table 1.4 that shows the relationship between security services and attacks.

**1.6** Draw a matrix similar to Table 1.4 that shows the relationship between security mechanisms and attacks.

**PART 1: CRYPTOGRAPHY**

**CHAPTER**

**SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

**2.1 Symmetric Encryption Principles**

Cryptography

Cryptanalysis

Feistel Cipher Structure

**2.2 Symmetric Block Encryption Algorithms**

Data Encryption Standard

Triple DES

Advanced Encryption Standard

**2.3 Random and Pseudorandom Numbers**

The Use of Random Numbers

TRNGs, PRNGs, and PRFs

Algorithm Design

**2.4 Stream Ciphers and RC4**

Stream Cipher Structure

The RC4 Algorithm

**2.5 Cipher Block Modes of Operation**

Electronic Codebook Mode

Cipher Block Chaining Mode

Cipher Feedback Mode

Counter Mode

**2.6 Recommended Reading and Web Sites**

**2.7 Key Terms, Review Questions, and Problems**

**27**

**28 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

*All the afternoon Mungo had been working on Stern’s code, principally with the aid of the latest messages which he had copied down at the Nevin Square drop. Stern was very confident. He must be well aware London Central knew about that drop. It was obvious that they didn’t care how often Mungo read their messages, so confident were they in the impenetrability of the code.*

—*Talking to Strange Men,* Ruth Rendell

*Amongst the tribes of Central Australia every man, woman, and child has a secret or sacred name which is bestowed by the older men upon him or her soon after birth, and which is known to none but the fully initiated members of the group.This secret name is never mentioned except upon the most solemn occasions; to utter it in the hearing of men of another group would be a most serious breach of tribal custom. When mentioned at all, the name is spoken only in a whisper, and not until the most elaborate precautions have been taken that it shall be heard by no one but members of the group. The native thinks that a stranger knowing his secret name would have special power to work him ill by means of magic.*

—*The Golden Bough,* Sir James George Frazer

Symmetric encryption, also referred to as conventional encryption, secret-key, or single-key encryption, was the only type of encryption in use prior to the develop ment of public-key encryption in the late 1970s.1 It remains by far the most widely used of the two types of encryption.

This chapter begins with a look at a general model for the symmetric encryp tion process; this will enable us to understand the context within which the algorithms are used. Then we look at three important block encryption algorithms: DES, triple DES, and AES. This is followed by a discussion of random and pseudorandom number generation. Next, the chapter introduces symmetric stream encryption and describes the widely used stream cipher RC4. Finally, we look at the important topic of block cipher modes of operation.

**2.1 SYMMETRIC ENCRYPTION PRINCIPLES**

A **symmetric encryption** scheme has five ingredients (Figure 2.1):

• **Plaintext:** This is the original message or data that is fed into the algorithm as input.

• **Encryption algorithm:**The encryption algorithm performs various substitutions and transformations on the plaintext.

• **Secret key:** The secret key is also input to the algorithm.The exact substitutions and transformations performed by the algorithm depend on the key.

1Public-key encryption was first described in the open literature in 1976; the National Security Agency (NSA) claims to have discovered it some years earlier.

**Secret key shared by sender and recipient**

***K***

***X***

**2.1 / SYMMETRIC ENCRYPTION PRINCIPLES 29**

**Secret key shared by**

**sender and recipient**

***K***

**Transmitted**

**ciphertext**

***Y***  **E[*K*, *X*] *X***  **D[*K*, *Y*]**

**Plaintext input**

**Encryption algorithm (e.g., AES)**

**Decryption algorithm (reverse of encryption algorithm)**

**Plaintext output**

**Figure 2.1** Simplified Model of Symmetric Encryption

• **Ciphertext:** This is the scrambled message produced as output. It depends on the plaintext and the secret key. For a given message, two different keys will produce two different ciphertexts.

• **Decryption algorithm:** This is essentially the encryption algorithm run in reverse. It takes the ciphertext and the same secret key and produces the orig inal plaintext.

There are two requirements for secure use of symmetric encryption:

**1.** We need a strong encryption algorithm. At a minimum, we would like the algorithm to be such that an opponent who knows the algorithm and has access to one or more ciphertexts would be unable to decipher the ciphertext or figure out the key.This requirement is usually stated in a stronger form:The opponent should be unable to decrypt ciphertext or discover the key even if he or she is in possession of a number of ciphertexts together with the plaintext that produced each ciphertext.

**2.** Sender and receiver must have obtained copies of the secret key in a secure fashion and must keep the key secure. If someone can discover the key and knows the algorithm, all communication using this key is readable.

It is important to note that the security of symmetric encryption depends on the secrecy of the key, not the secrecy of the algorithm.That is, it is assumed that it is impractical to decrypt a message on the basis of the ciphertext *plus* knowledge of the encryption/decryption algorithm. In other words, we do not need to keep the algorithm secret; we need to keep only the key secret.

This feature of symmetric encryption is what makes it feasible for widespread use. The fact that the algorithm need not be kept secret means that manufacturers can and have developed low-cost chip implementations of data encryption algo rithms. These chips are widely available and incorporated into a number of products. With the use of symmetric encryption, the principal security problem is maintaining the secrecy of the key.

**30 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY Cryptography**

Cryptographic systems are generically classified along three independent dimensions:

**1. The type of operations used for transforming plaintext to ciphertext.** All encryption algorithms are based on two general principles: substitution, in which each element in the plaintext (bit, letter, group of bits or letters) is mapped into another element, and transposition, in which elements in the plaintext are rearranged. The fundamental requirement is that no information be lost (that is, that all operations be reversible). Most systems, referred to as product systems, involve multiple stages of substitutions and transpositions.

**2. The number of keys used.** If both sender and receiver use the same key, the system is referred to as symmetric, single-key, secret-key, or conventional encryption. If the sender and receiver each use a different key,the system is referred to as asymmetric, two-key, or public-key encryption.

**3. The way in which the plaintext is processed.** A **block cipher** processes the input one block of elements at a time, producing an output block for each input block. A **stream cipher** processes the input elements continuously, producing output one element at a time, as it goes along.

**Cryptanalysis**

The process of attempting to discover the plaintext or key is known as **cryptanalysis**. The strategy used by the cryptanalyst depends on the nature of the encryption scheme and the information available to the cryptanalyst.

Table 2.1 summarizes the various types of cryptanalytic attacks based on the amount of information known to the cryptanalyst. The most difficult problem is presented when all that is available is the *ciphertext only*. In some cases, not even the encryption algorithm is known, but in general, we can assume that the opponent does know the algorithm used for encryption. One possible attack under these circumstances is the brute-force approach of trying all possible keys. If the key space is very large, this becomes impractical. Thus, the opponent must rely on an analysis of the ciphertext itself, generally applying various statistical tests to it. To use this approach, the opponent must have some general idea of the type of plaintext that is concealed, such as English or French text, an EXE file, a Java source listing, an accounting file, and so on.

The ciphertext-only attack is the easiest to defend against because the opponent has the least amount of information to work with. In many cases, however, the analyst has more information. The analyst may be able to capture one or more plaintext messages as well as their encryptions. Or the analyst may know that certain plain text patterns will appear in a message. For example, a file that is encoded in the Postscript format always begins with the same pattern, or there may be a standardized header or banner to an electronic funds transfer message, and so on. All of these are examples of *known plaintext*. With this knowledge, the analyst may be able to deduce the key on the basis of the way in which the known plaintext is transformed.

Closely related to the known-plaintext attack is what might be referred to as a probable-word attack. If the opponent is working with the encryption of some general

**2.1 / SYMMETRIC ENCRYPTION PRINCIPLES 31**

**Table 2.1** Types of Attacks on Encrypted Messages

**Type of Attack Known to Cryptanalyst**

Ciphertext only • Encryption algorithm

• Ciphertext to be decoded

Known plaintext • Encryption algorithm

• Ciphertext to be decoded

• One or more plaintext–ciphertext pairs formed with the secret key

Chosen plaintext • Encryption algorithm

• Ciphertext to be decoded

• Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext

generated with the secret key

Chosen ciphertext • Encryption algorithm

• Ciphertext to be decoded

• Purported ciphertext chosen by cryptanalyst, together with its corresponding

decrypted plaintext generated with the secret key

Chosen text • Encryption algorithm

• Ciphertext to be decoded

• Plaintext message chosen by cryptanalyst, together with its corresponding

ciphertext generated with the secret key

• Purported ciphertext chosen by cryptanalyst, together with its corresponding

decrypted plaintext generated with the secret key

prose message, he or she may have little knowledge of what is in the message. However, if the opponent is after some very specific information, then parts of the message may be known. For example, if an entire accounting file is being transmitted, the opponent may know the placement of certain key words in the header of the file. As another example, the source code for a program developed by a corporation might include a copyright statement in some standardized position.

If the analyst is able somehow to get the source system to insert into the system a message chosen by the analyst, then a *chosen-plaintext* attack is possible. In general, if the analyst is able to choose the messages to encrypt, the analyst may deliberately pick patterns that can be expected to reveal the structure of the key.

Table 2.1 lists two other types of attack: chosen ciphertext and chosen text. These are less commonly employed as cryptanalytic techniques but are nevertheless possible avenues of attack.

Only relatively weak algorithms fail to withstand a ciphertext-only attack. Generally, an encryption algorithm is designed to withstand a known-plaintext attack. An encryption scheme is **computationally secure** if the ciphertext generated by the scheme meets one or both of the following criteria:

• The cost of breaking the cipher exceeds the value of the encrypted information. • The time required to break the cipher exceeds the useful lifetime of the information.

**32 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

Unfortunately, it is very difficult to estimate the amount of effort required to cryptanalyze ciphertext successfully. However, assuming there are no inherent mathematical weaknesses in the algorithm, then a brute-force approach is indicated, and here we can make some reasonable estimates about costs and time.

A brute-force approach involves trying every possible key until an intelligi ble translation of the ciphertext into plaintext is obtained. On average, half of all possible keys must be tried to achieve success. Table 2.2 shows how much time is involved for various key sizes. The 56-bit key size is used with the DES (Data Encryption Standard) algorithm. For each key size, the results are shown assuming that it takes 1 *µ*s to perform a single decryption, which is a reasonable order of magnitude for today’s machines. With the use of massively parallel organizations of microprocessors, it may be possible to achieve processing rates many orders of magnitude greater. The final column of Table 2.2 considers the results for a system that can process 1 million keys per microsecond. As you can see, at this performance level, DES no longer can be considered computationally secure.

**Feistel Cipher Structure**

Many symmetric block encryption algorithms, including DES, have a structure first described by Horst Feistel of IBM in 1973 [FEIS73] and shown in Figure 2.2. The inputs to the encryption algorithm are a plaintext block of length 2*w* bits and a key *K*. The plaintext block is divided into two halves, *LE*0 and *RE*0. The two halves of the data pass through *n* rounds of processing and then combine to produce the ciphertext block. Each round *i* has as inputs *LEi* 1 and *REi* 1 derived from the

- -

previous round, as well as a subkey *Ki* derived from the overall *K*. In general, the subkeys *Ki* are different from *K* and from each other and are generated from the key by a subkey generation algorithm. In Figure 2.2, 16 rounds are used, although any number of rounds could be implemented. The right-hand side of Figure 2.2 shows the decryption process.

All rounds have the same structure. A substitution is performed on the left half of the data.This is done by applying a *round function* F to the right half of the data and then taking the exclusive-OR (XOR) of the output of that function and the left half of the data. The round function has the same general structure for each round but is

**Table 2.2** Average Time Required for Exhaustive Key Search

**Key Size (bits)**

**Number of**

**Alternative Keys**

**Time Required at 1 Decryption/*µ*s**

**Time Required at 106 Decryptions/*µ*s**

32 232 = 4.3 \* 109 231ms = 35.8 minutes 2.15 milliseconds 56 256 = 7.2 \* 1016 255ms = 1142 years 10.01 hours 128 2128 = 3.4 \* 1038 2127ms = 5.4 \* 1024 years 5.4 \* 1018 years 168 2168 = 3.7 \* 1050 2167ms = 5.9 \* 1036 years 5.9 \* 1030 years

26 characters

(permutation) 26! = 4 \* 1026 2 \* 1026ms = 6.4 \* 1012 years 6.4 \* 106 years

**2.1 / SYMMETRIC ENCRYPTION PRINCIPLES 33 Output (plaintext)**

**1**

**d**

**n**

**u**

**o**

**R**

**Input (plaintext) *LE*0 *RE*0**

**F**

***K*1**

***RD*17 = *LE*0**

***LD*16 = *RE*0**

**6**

**1**

**d**

**n**

**u**

**o**

**R**

***LD*17 = *RE*0 *RD*16 = *LE*0**

**F**

***K*1**

**2**

**dn**

**u**

**o**

**R**

**5**

**1**

**dn**

**u**

**o**

**R**

**6**

**1**

**dn**

**u**

**o**

**R**

***LE*1 *RE*1 F**

***LE*2 *RE*2**

***LE*14 *RE*14 F**

***LE*15 *RE*15 F**

***LE*16 *RE*16**

***LE*17 *RE*17 Output (ciphertext)**

***K*2**

***K*15 *K*16**

**5**

**1**

**dn**

**u**

**o**

**R**

**2**

**dn**

**u**

**o**

**R**

**1**

**dn**

**u**

**o**

**R**

***RD*15 *LD*15 = *LE*1 = *RE*1**

**F**

***LD*14 = *RE*2 *RD*14 = *LE*2**

***LD*2 = *RE*14 *RD*2 = *LE*14**

**F**

***RD*1 *LD*1 = *LE*15 = *RE*15**

**F**

***LD*0 = *RE*16 *RD*0 = *LE*16 Input (ciphertext)**

***K*2**

***K*15 *K*16**

**Figure 2.2** Feistel Encryption and Decryption (16 rounds)

parameterized by the round subkey *Ki*. Following this substitution, a permutation is performed that consists of the interchange of the two halves of the data. The **Feistel structure** is a particular example of the more general structure used by all symmetric block ciphers. In general, a symmetric block cipher consists of a sequence of rounds, with each round performing substitutions and permutations conditioned by a secret key value. The exact realization of a symmetric block cipher depends on the choice of the following parameters and design features.

**34 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

• **Block size:** Larger block sizes mean greater security (all other things being equal) but reduced encryption/decryption speed. A block size of 128 bits is a reasonable tradeoff and is nearly universal among recent block cipher designs.

• **Key size:** Larger key size means greater security but may decrease encryption/ decryption speed.The most common key length in modern algorithms is 128 bits. • **Number of rounds:** The essence of a symmetric block cipher is that a single round offers inadequate security but that multiple rounds offer increasing security. A typical size is 16 rounds.

• **Subkey generation algorithm:** Greater complexity in this algorithm should lead to greater difficulty of cryptanalysis.

• **Round function:** Again, greater complexity generally means greater resistance to cryptanalysis.

There are two other considerations in the design of a symmetric block cipher:

• **Fast software encryption/decryption:** In many cases, encryption is embedded in applications or utility functions in such a way as to preclude a hardware implementation. Accordingly, the speed of execution of the algorithm becomes a concern.

• **Ease of analysis:** Although we would like to make our algorithm as difficult as possible to cryptanalyze, there is great benefit in making the algorithm easy to analyze. That is, if the algorithm can be concisely and clearly explained, it is easier to analyze that algorithm for cryptanalytic vulnerabilities and therefore develop a higher level of assurance as to its strength. DES, for example, does not have an easily analyzed functionality.

Decryption with a symmetric block cipher is essentially the same as the encryp tion process. The rule is as follows: Use the ciphertext as input to the algorithm, but use the subkeys *Ki* in reverse order. That is, use *Kn* in the first round, *Kn* 1 in the

-

second round, and so on until *K*1 is used in the last round. This is a nice feature, because it means we need not implement two different algorithms—one for encryp tion and one for decryption.

**2.2 SYMMETRIC BLOCK ENCRYPTION ALGORITHMS**

The most commonly used symmetric encryption algorithms are block ciphers. A **block cipher** processes the plaintext input in fixed-sized blocks and produces a block of ciphertext of equal size for each plaintext block.This section focuses on the three most important symmetric block ciphers: the Data Encryption Standard (DES), triple DES (3DES), and the Advanced Encryption Standard (AES).

**Data Encryption Standard**

The most widely used encryption scheme is based on the **Data Encryption Standard (DES)** issued in 1977, as Federal Information Processing Standard 46 (FIPS 46) by the National Bureau of Standards, now known as the National Institute of Standards

**2.2 / SYMMETRIC BLOCK ENCRYPTION ALGORITHMS 35**

and Technology (NIST). The algorithm itself is referred to as the Data Encryption Algorithm (DEA).2

***DESCRIPTION OF THE ALGORITHM*** The plaintext is 64 bits in length and the key is 56 bits in length; longer plaintext amounts are processed in 64-bit blocks. The DES structure is a minor variation of the Feistel network shown in Figure 2.2. There are 16 rounds of processing. From the original 56-bit key, 16 subkeys are generated, one of which is used for each round.

The process of decryption with DES is essentially the same as the encryption process.The rule is as follows: Use the ciphertext as input to the DES algorithm, but use the subkeys *Ki* in reverse order. That is, use *K*16 on the first iteration, *K*15 on the second iteration, and so on until *K*1 is used on the 16th and last iteration.

***THE STRENGTH OF DES*** Concerns about the strength of DES fall into two categories: concerns about the algorithm itself and concerns about the use of a 56-bit key. The first concern refers to the possibility that cryptanalysis is possible by exploiting the characteristics of the DES algorithm. Over the years, there have been numerous attempts to find and exploit weaknesses in the algorithm, making DES the most-studied encryption algorithm in existence. Despite numerous approaches,

no one has so far succeeded in discovering a fatal weakness in DES.3 A more serious concern is key length.With a key length of 56 bits, there are 256 possible keys, which is approximately 7.2 ⋅ 1016 keys. Thus, on the face of it, a brute force attack appears impractical.Assuming that on average half the key space has to be searched, a single machine performing one DES encryption per microsecond would take more than a thousand years (see Table 2.2) to break the cipher. However, the assumption of one encryption per microsecond is overly conserv ative. DES finally and definitively proved insecure in July 1998, when the Electronic Frontier Foundation (EFF) announced that it had broken a DES encryption using a special-purpose “DES cracker” machine that was built for less than $250,000. The attack took less than three days. The EFF has published a detailed description of the machine, enabling others to build their own cracker [EFF98]. And, of course, hard ware prices will continue to drop as speeds increase, making DES virtually worthless. It is important to note that there is more to a key-search attack than simply running through all possible keys. Unless known plaintext is provided, the analyst must be able to recognize plaintext as plaintext. If the message is just plain text in English, then the result pops out easily, although the task of recognizing English would have to be automated. If the text message has been compressed before encryption, then recognition is more difficult. And if the message is some more gen eral type of data, such as a numerical file, and this has been compressed, the problem becomes even more difficult to automate. Thus, to supplement the brute-force

2The terminology is a bit confusing. Until recently, the terms *DES* and *DEA* could be used interchange ably. However, the most recent edition of the DES document includes a specification of the DEA described here plus the triple DEA (3DES) described subsequently. Both DEA and 3DES are part of the Data Encryption Standard. Furthermore, until the recent adoption of the official term *3DES*, the triple DEA algorithm was typically referred to as *triple DES* and written as 3DES. For the sake of convenience, we will use 3DES.

3At least, no one has publicly acknowledged such a discovery.

**36 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

approach, some degree of knowledge about the expected plaintext is needed, and some means of automatically distinguishing plaintext from garble is also needed.The EFF approach addresses this issue as well and introduces some automated tech niques that would be effective in many contexts.

A final point: If the only form of attack that could be made on an encryption algorithm is brute force, then the way to counter such attacks is obvious: use longer keys. To get some idea of the size of key required, let us use the EFF cracker as a basis for our estimates. The EFF cracker was a prototype, and we can assume that with today’s technology a faster machine is cost effective. If we assume that a cracker can perform one million decryptions per *µ*s, which is the rate used in Table 2.2, then a DES code would take about 10 hours to crack. This is a speed-up of approximately a factor of 7 compared to the EFF result. Using this rate, Figure 2.3 shows how long it would take to crack a DES-style algorithm as a function of key size. For example, for a 128-bit key, which is common among contemporary algorithms, it would take over 1018 years to break the code using the EFF cracker. Even if we managed to speed up the cracker by a factor of 1 trillion (1012), it would still take over 1 million years to break the code. So a 128-bit key is guaranteed to result in an algorithm that is unbreakable by brute force.

**Triple DES**

Triple DES (3DES) was first standardized for use in financial applications in ANSI standard X9.17 in 1985. 3DES was incorporated as part of the Data Encryption Standard in 1999 with the publication of FIPS 46-3.

**1044**

**1040**

**1036**

**1032**

**1028**

**1024**

**k**

**a**

**e**

**r**

**1020**

**B**

**o**

**t**

**s**

**r**

**1016**

**a**

**e**

**Y**

**1012**

**108**

**104**

**100**

**10****4**

**50 100 128 150 168 200**

**56**

**Key Length (bits)**

**Figure 2.3** Time to Break a Code (assuming 106 decryptions/ s) m

**2.2 / SYMMETRIC BLOCK ENCRYPTION ALGORITHMS 37**

***K*1**

***K*2**

***K*3**

***P A B C***

**E**

***K*3**

**D**

**(a) Encryption *K*2**

**E**

***K*1**

***C B A P***

**D**

**Figure 2.4** Triple DES

**E**

**(b) Decryption**

**D**

3DES uses three keys and three executions of the DES algorithm. The function follows an encrypt-decrypt-encrypt (EDE) sequence (Figure 2.4a):

=

*C* E(*K*3, D(*K*2, E(*K*1, *P*)))

where

=

*C* ciphertext

=

*P* plaintext

=

E[*K*, *X*] encryption of *X* using key *K*

=

D[*K*, *Y* ] decryption of *Y* using key *K*

Decryption is simply the same operation with the keys reversed (Figure 2.4b): *P* D(*K*1, E(*K*2, D(*K*3, *C* )))

=

There is no cryptographic significance to the use of decryption for the second stage of 3DES encryption. Its only advantage is that it allows users of 3DES to decrypt data encrypted by users of the older single DES:

*C* E(*K*1, D(*K*1, E(*K*1, *P*))) E[*K*, *P*]

= =

With three distinct keys, 3DES has an effective key length of 168 bits. FIPS 46-3 also allows for the use of two keys, with *K*1 *K*3; this provides for a key length of 112

=

bits. FIPS 46-3 includes the following guidelines for 3DES.

• 3DES is the FIPS approved symmetric encryption algorithm of choice. • The original DES, which uses a single 56-bit key, is permitted under the standard for legacy systems only. New procurements should support 3DES. • Government organizations with legacy DES systems are encouraged to transition to 3DES.

• It is anticipated that 3DES and the Advanced Encryption Standard (AES) will coexist as FIPS-approved algorithms, allowing for a gradual transition to AES.

It is easy to see that 3DES is a formidable algorithm. Because the underlying cryptographic algorithm is DEA, 3DES can claim the same resistance to cryptanalysis

**38 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

based on the algorithm as is claimed for DEA. Furthermore, with a 168-bit key length, brute-force attacks are effectively impossible.

Ultimately,AES is intended to replace 3DES, but this process will take a number of years. NIST anticipates that 3DES will remain an approved algorithm (for U.S. government use) for the foreseeable future.

**Advanced Encryption Standard**

3DES has two attractions that assure its widespread use over the next few years. First, with its 168-bit key length, it overcomes the vulnerability to brute-force attack of DEA. Second, the underlying encryption algorithm in 3DES is the same as in DEA.This algorithm has been subjected to more scrutiny than any other encryption algorithm over a longer period of time, and no effective cryptanalytic attack based on the algorithm rather than brute force has been found.Accordingly, there is a high level of confidence that 3DES is very resistant to cryptanalysis. If security were the only consideration, then 3DES would be an appropriate choice for a standardized encryption algorithm for decades to come.

The principal drawback of 3DES is that the algorithm is relatively sluggish in software. The original DEA was designed for mid-1970s hardware implementation and does not produce efficient software code. 3DES, which has three times as many rounds as DEA, is correspondingly slower.A secondary drawback is that both DEA and 3DES use a 64-bit block size. For reasons of both efficiency and security, a larger block size is desirable.

Because of these drawbacks, 3DES is not a reasonable candidate for long-term use. As a replacement, NIST in 1997 issued a call for proposals for a new **Advanced Encryption Standard (AES)**, which should have a security strength equal to or better than 3DES and significantly improved efficiency. In addition to these general requirements, NIST specified that AES must be a symmetric block cipher with a block length of 128 bits and support for key lengths of 128, 192, and 256 bits. Evaluation criteria included security, computational efficiency, memory require ments, hardware and software suitability, and flexibility.

In a first round of evaluation, 15 proposed algorithms were accepted.A second round narrowed the field to five algorithms. NIST completed its evaluation process and published a final standard (FIPS PUB 197) in November of 2001. NIST selected Rijndael as the proposed AES algorithm. The two researchers who developed and submitted Rijndael for the AES are both cryptographers from Belgium: Dr. Joan Daemen and Dr. Vincent Rijmen.

***OVERVIEW OF THE ALGORITHM*** AES uses a block length of 128 bits and a key length that can be 128, 192, or 256 bits. In the description of this section, we assume a key length of 128 bits, which is likely to be the one most commonly implemented.

Figure 2.5 shows the overall structure of AES. The input to the encryption and decryption algorithms is a single 128-bit block. In FIPS PUB 197, this block is depicted as a square matrix of bytes.This block is copied into the **State** array, which is modified at each stage of encryption or decryption. After the final stage, **State** is copied to an output matrix. Similarly, the 128-bit key is depicted as a square matrix of bytes. This key is then expanded into an array of key schedule words: each word is four bytes and the total key schedule is 44 words for the 128-bit key. The ordering of bytes within a

**2.2 / SYMMETRIC BLOCK ENCRYPTION ALGORITHMS 39 Key**

**1**

**dn**

**u**

**o**

**R**

**9**

**dn**

**u**

**o**

**R**

**0**

**1**

**dn**

**u**

**o**

**R**

**Plaintext**

**(16 bytes)**

**Add round key Substitute bytes Shift rows**

**Mix columns Add round key**

**•**

**•**

**•**

**Substitute bytes Shift rows**

**Mix columns Add round key Substitute bytes Shift rows**

**Add round key**

**Ciphertext**

**(16 bytes)**

**(a) Encryption**

**(16 bytes) Expand key**

**w[0, 3]**

**w[4, 7]**

**w[36, 39] w[40, 43]**

**Plaintext**

**(16 bytes)**

**Add round key Inverse sub bytes Inverse shift rows**

**Inverse mix cols Add round key Inverse sub bytes Inverse shift rows**

**•**

**•**

**•**

**Inverse mix cols Add round key Inverse sub bytes Inverse shift rows Add round key**

**Ciphertext**

**(16 bytes)**

**(b) Decryption**

**0**

**1**

**d**

**n**

**u**

**o**

**R**

**9**

**dn**

**u**

**o**

**R**

**1**

**dn**

**u**

**o**

**R**

**Figure 2.5** AES Encryption and Decryption

matrix is by column. So, for example, the first four bytes of a 128-bit plaintext input to the encryption cipher occupy the first column of the **in** matrix, the second four bytes occupy the second column, and so on. Similarly, the first four bytes of the expanded key, which form a word, occupy the first column of the **w** matrix.

The following comments give some insight into AES.

**1.** One noteworthy feature of this structure is that it is not a Feistel structure. Recall that in the classic Feistel structure, half of the data block is used to modify the other half of the data block, and then the halves are swapped. AES does not use

**40 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

a Feistel structure but processes the entire data block in parallel during each round using substitutions and permutation.

**2.** The key that is provided as input is expanded into an array of forty-four 32-bit words, **w**[*i*]. Four distinct words (128 bits) serve as a round key for each round. **3.** Four different stages are used, one of permutation and three of substitution: • **Substitute bytes:** Uses a table, referred to as an S-box,4 to perform a byte-by byte substitution of the block.

• **Shift rows:** A simple permutation that is performed row by row.

• **Mix columns:** A substitution that alters each byte in a column as a function of all of the bytes in the column.

• **Add round key:** A simple bitwise XOR of the current block with a portion of the expanded key.

**4.** The structure is quite simple. For both encryption and decryption, the cipher begins with an Add Round Key stage, followed by nine rounds that each includes all four stages, followed by a tenth round of three stages. Figure 2.6 depicts the structure of a full encryption round.

**5.** Only the Add Round Key stage makes use of the key. For this reason, the cipher begins and ends with an Add Round Key stage. Any other stage, applied at the beginning or end, is reversible without knowledge of the key and so would add no security.

**6.** The Add Round Key stage by itself would not be formidable. The other three stages together scramble the bits, but by themselves, they would provide no secu rity because they do not use the key. We can view the cipher as alternating oper ations of XOR encryption (Add Round Key) of a block, followed by scrambling of the block (the other three stages), followed by XOR encryption, and so on. This scheme is both efficient and highly secure.

**7.** Each stage is easily reversible. For the Substitute Byte, Shift Row, and Mix Columns stages, an inverse function is used in the decryption algorithm. For the Add Round Key stage, the inverse is achieved by XORing the same round key to the block, using the result that A B B A.

=

**8.** As with most block ciphers, the decryption algorithm makes use of the expanded key in reverse order. However, the decryption algorithm is not identical to the encryption algorithm. This is a consequence of the particular structure of AES.

**9.** Once it is established that all four stages are reversible, it is easy to verify that decryption does recover the plaintext. Figure 2.5 lays out encryption and decryp tion going in opposite vertical directions. At each horizontal point (e.g., the dashed line in the figure), **State** is the same for both encryption and decryption.

**10.** The final round of both encryption and decryption consists of only three stages. Again, this is a consequence of the particular structure of AES and is required to make the cipher reversible.

4The term *S-box*, or substitution box, is commonly used in the description of symmetric ciphers to refer to a table used for a table-lookup type of substitution mechanism.

**1**

***r***

**S**

**e**

**t**

**a**

**tS**

**S S S S S S S S S S S**

**s**

**e**

**t**

**y**

**B**

**b**

**u**

**S**

**e**

**t**

**a**

**tS**

**s**

**w**

**o**

**Rt**

**fi**

**hS**

**e**

**t**

**a**

**tS**

**M M**

**M**

**s**

**n**

**m**

**u**

**l**

**o**

**C**

**x**

**i**

**M**

**e**

**t**

**a**

**tS**

**1**

**1**

***r***

**0**

**1**

***r***

**9**

***r***

**8**

***r***

**7**

***r***

**6**

***r***

**5**

***r***

**4**

***r***

**3**

***r***

**2**

***r***

**1**

***r***

**0**

***r***

**y**

**e**

**Kd**

**n**

**u**

**o**

**R**

**dd**

**A**

**e**

**t**

**a**

**tS**

d

n

u

o

R

n

o

i

t

p

y

r

c

n

E

S

E

A

**6**

**.**

**2**

**e**

**r**

**u**

**g**

**i**

**F**

**41**

**42 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY 2.3 RANDOM AND PSEUDORANDOM NUMBERS**

Random numbers play an important role in the use of encryption for various network security applications.We provide an overview in this section.The topic is examined in more detail in Appendix E.

**The Use of Random Numbers**

A number of network security algorithms based on cryptography make use of random numbers. For example,

• Generation of keys for the RSA public-key encryption algorithm (described in Chapter 3) and other public-key algorithms.

• Generation of a stream key for symmetric stream cipher (discussed in the following section).

• Generation of a symmetric key for use as a temporary session key.This function is used in a number of networking applications, such as Transport Layer Security (Chapter 5), Wi-Fi (Chapter 6), e-mail security (Chapter 7), and IP security (Chapter 8).

• In a number of key distribution scenarios, such as Kerberos (Chapter 4), random numbers are used for handshaking to prevent replay attacks.

These applications give rise to two distinct and not necessarily compatible requirements for a sequence of random numbers: randomness and unpredictability.

***RANDOMNESS*** Traditionally, the concern in the generation of a sequence of allegedly random numbers has been that the sequence of numbers be random in some well defined statistical sense. The following criteria are used to validate that a sequence of numbers is random.

• **Uniform distribution:**The distribution of bits in the sequence should be uniform; that is, the frequency of occurrence of ones and zeros should be approximately the same.

• **Independence:** No one subsequence in the sequence can be inferred from the others.

Although there are well-defined tests for determining that a sequence of num bers matches a particular distribution, such as the uniform distribution, there is no such test to “prove” independence. Rather, a number of tests can be applied to demonstrate if a sequence does not exhibit independence.The general strategy is to apply a number of such tests until the confidence that independence exists is sufficiently strong.

In the context of our discussion, the use of a sequence of numbers that appear statistically random often occurs in the design of algorithms related to cryptography. For example, a fundamental requirement of the RSA public-key encryption scheme discussed in Chapter 3 is the ability to generate prime numbers. In general, it is difficult to determine if a given large number *N* is prime. A brute-force approach 1*N*

would be to divide *N* by every odd integer less than . If *N* is on the order, say, of 10150 (a not uncommon occurrence in public-key cryptography), such a brute-force

**2.3 / RANDOM AND PSEUDORANDOM NUMBERS 43**

approach is beyond the reach of human analysts and their computers. However, a number of effective algorithms exist that test the primality of a number by using a sequence of randomly chosen integers as input to relatively simple computations. If 210150

the sequence is sufficiently long (but far, far less than ), the primality of a number can be determined with near certainty.This type of approach, known as ran domization, crops up frequently in the design of algorithms. In essence, if a problem is too hard or time-consuming to solve exactly, a simpler, shorter approach based on randomization is used to provide an answer with any desired level of confidence.

***UNPREDICTABILITY*** In applications such as reciprocal authentication and session key generation, the requirement is not so much that the sequence of numbers be statistically random but that the successive members of the sequence are unpredictable. With “true” random sequences, each number is statistically independent of other numbers in the sequence and therefore unpredictable. However, as is discussed shortly, true random numbers are not always used; rather, sequences of numbers that appear to be random are generated by some algorithm. In this latter case, care must be taken that an opponent not be able to predict future elements of the sequence on the basis of earlier elements.

**TRNGs, PRNGs, and PRFs**

Cryptographic applications typically make use of algorithmic techniques for random number generation. These algorithms are deterministic and therefore produce sequences of numbers that are not statistically random. However, if the algorithm is good, the resulting sequences will pass many reasonable tests of randomness. Such numbers are referred to as **pseudorandom numbers**.

You may be somewhat uneasy about the concept of using numbers generated by a deterministic algorithm as if they were random numbers. Despite what might be called “philosophical” objections to such a practice, it generally works. As one expert on probability theory puts it [HAMM91],

For practical purposes we are forced to accept the awkward concept of “relatively random” meaning that with regard to the proposed use we can see no reason why they will not perform as if they were random (as the theory usually requires).This is highly subjective and is not very palatable to purists, but it is what statisticians regularly appeal to when they take “a random sample”—they hope that any results they use will have approximately the same properties as a complete counting of the whole sample space that occurs in their theory.

Figure 2.7 contrasts a **true random number generator (TRNG)** with two forms of pseudorandom number generators. A TRNG takes as input a source that is effectively random; the source is often referred to as an **entropy source**. In essence, the entropy source is drawn from the physical environment of the com puter and could include things such as keystroke timing patterns, disk electrical activity, mouse movements, and instantaneous values of the system clock. The source, or combination of sources, serves as input to an algorithm that produces

**44 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

**Source of true**

**randomness**

**Seed**

**Seed**

**Context specific values**

**Conversion to binary**

**Random**

**bit stream (a) TRNG**

**Deterministic algorithm**

**Pseudorandom bit stream**

**(b) PRNG**

**Deterministic algorithm**

**Pseudorandom value**

**(c) PRF**

**TRNG = true random number generator**

**PRNG = pseudorandom number generator**

**PRF = pseudorandom function**

**Figure 2.7** Random and Pseudorandom Number Generators

random binary output. The TRNG may simply involve conversion of an analog source to a binary output. The TRNG may involve additional processing to over come any bias in the source.

In contrast, a PRNG takes as input a fixed value, called the **seed**, and produces a sequence of output bits using a deterministic algorithm.Typically, as shown in Figure 2.7, there is some feedback path by which some of the results of the algorithm are fed back as input as additional output bits are produced. The important thing to note is that the output bit stream is determined solely by the input value or values, so that an adversary who knows the algorithm and the seed can reproduce the entire bit stream.

Figure 2.7 shows two different forms of PRNGs, based on application.

• **Pseudorandom number generator:** An algorithm that is used to produce an open-ended sequence of bits is referred to as a PRNG. A common application for an open-ended sequence of bits is as input to a symmetric stream cipher, as discussed in the following section.

• **Pseudorandom function (PRF):** A PRF is used to produce a pseudorandom string of bits of some fixed length. Examples are symmetric encryption keys and nonces. Typically, the PRF takes as input a seed plus some context specific values, such as a user ID or an application ID. A number of examples of PRFs will be seen throughout this book.

Other than the number of bits produced, there is no difference between a PRNG and a PRF. The same algorithms can be used in both applications. Both require a seed and both must exhibit randomness and unpredictability. Furthermore, a PRNG application may also employ context-specific input.

**2.4 / STREAM CIPHERS AND RC4 45**

**Algorithm Design**

Cryptographic PRNGs have been the subject of much research over the years, and a wide variety of algorithms have been developed.These fall roughly into two categories:

• **Purpose-built algorithms:** These are algorithms designed specifically and solely for the purpose of generating pseudorandom bit streams. Some of these algo rithms are used for a variety of PRNG applications; several of these are described in the next section. Others are designed specifically for use in a stream cipher. The most important example of the latter is RC4, described in the next section.

• **Algorithms based on existing cryptographic algorithms:** Cryptographic algo rithms have the effect of randomizing input. Indeed, this is a requirement of such algorithms. For example, if a symmetric block cipher produced ciphertext that had certain regular patterns in it, it would aid in the process of cryptanalysis. Thus, cryptographic algorithms can serve as the core of PRNGs. Three broad categories of cryptographic algorithms are commonly used to create PRNGs:

—**Symmetric block ciphers**

—**Asymmetric ciphers**

—**Hash functions and message authentication codes**

Any of these approaches can yield a cryptographically strong PRNG. A purpose-built algorithm may be provided by an operating system for general use. For applications that already use certain cryptographic algorithms for encryption or authentication, it makes sense to re-use the same code for the PRNG. Thus, all of these approaches are in common use.

**2.4 STREAM CIPHERS AND RC4**

A *block cipher* processes the input one block of elements at a time, producing an output block for each input block. A *stream cipher* processes the input elements continuously, producing output one element at a time as it goes along. Although block ciphers are far more common, there are certain applications in which a stream cipher is more appropriate. Examples are given subsequently in this book. In this section, we look at perhaps the most popular symmetric stream cipher, RC4. We begin with an overview of stream cipher structure, and then examine RC4.

**Stream Cipher Structure**

A typical stream cipher encrypts plaintext one byte at a time, although a stream cipher may be designed to operate on one bit at a time or on units larger than a byte at a time. Figure 2.8 is a representative diagram of stream cipher structure. In this structure, a key is input to a pseudorandom bit generator that produces a stream of 8-bit numbers that are apparently random. A pseudorandom stream is one that is unpredictable without knowledge of the input key and which has an apparently random character. The output of the generator, called a **keystream**, is combined one byte at a time with the plaintext stream using the bitwise exclusive-OR (XOR)

**46 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

**Key**

***K***

**Pseudorandom byte**

**generator**

**(key stream generator)**

**Plaintext**

**Key**

***K***

**Pseudorandom byte**

**generator**

**(key stream generator)**

**Ciphertext**

**Plaintext**

**byte stream *M***

***k***

**ENCRYPTION**

**byte stream *C***

***k***

**DECRYPTION**

**byte stream *M***

**Figure 2.8** Stream Cipher Diagram

operation. For example, if the next byte generated by the generator is 01101100 and the next plaintext byte is 11001100, then the resulting ciphertext byte is

11001100 plaintext

01101100 key stream 10100000 ciphertext

Decryption requires the use of the same pseudorandom sequence: 10100000 ciphertext

01101100 key stream 11001100 plaintext

[KUMA97] lists the following important design considerations for a stream cipher.

**1.** The encryption sequence should have a large period. A pseudorandom num ber generator uses a function that produces a deterministic stream of bits that eventually repeats.The longer the period of repeat, the more difficult it will be to do cryptanalysis.

**2.** The keystream should approximate the properties of a true random number stream as close as possible. For example, there should be an approximately equal number of 1s and 0s. If the keystream is treated as a stream of bytes, then all of the 256 possible byte values should appear approximately equally often. The more random-appearing the keystream is, the more randomized the ciphertext is, making cryptanalysis more difficult.

**3.** Note from Figure 2.8 that the output of the pseudorandom number generator is conditioned on the value of the input key. To guard against brute-force attacks, the key needs to be sufficiently long.The same considerations as apply for block ciphers are valid here. Thus, with current technology, a key length of at least 128 bits is desirable.

**2.4 / STREAM CIPHERS AND RC4 47**

With a properly designed pseudorandom number generator, a stream cipher can be as secure as block cipher of comparable key length.The primary advantage of a stream cipher is that stream ciphers are almost always faster and use far less code than do block ciphers.The example in this section, RC4, can be implemented in just a few lines of code. Table 2.3, using data from [RESC01], compares execution times of RC4 with three well-known symmetric block ciphers. The advantage of a block cipher is that you can reuse keys. However, if two plaintexts are encrypted with the same key using a stream cipher, then cryptanalysis is often quite simple [DAWS96]. If the two ciphertext streams are XORed together, the result is the XOR of the original plaintexts. If the plaintexts are text strings, credit card numbers, or other byte streams with known properties, then cryptanalysis may be successful.

For applications that require encryption/decryption of a stream of data (such as over a data-communications channel or a browser/Web link), a stream cipher might be the better alternative. For applications that deal with blocks of data (such as file transfer, e-mail, and database), block ciphers may be more appropriate. However, either type of cipher can be used in virtually any application.

**The RC4 Algorithm**

RC4 is a stream cipher designed in 1987 by Ron Rivest for RSA Security. It is a vari able key-size stream cipher with byte-oriented operations. The algorithm is based on the use of a random permutation. Analysis shows that the period of the cipher is overwhelmingly likely to be greater than 10100 [ROBS95a]. Eight to sixteen machine operations are required per output byte, and the cipher can be expected to run very quickly in software. RC4 is used in the Secure Sockets Layer/Transport Layer Security (SSL/TLS) standards that have been defined for communication between Web browsers and servers. It is also used in the Wired Equivalent Privacy (WEP) protocol and the newer WiFi Protected Access (WPA) protocol that are part of the IEEE 802.11 wireless LAN standard. RC4 was kept as a trade secret by RSA Security. In September 1994, the RC4 algorithm was anonymously posted on the Internet on the Cypherpunks anonymous remailers list.

The RC4 algorithm is remarkably simple and quite easy to explain. A variable length key of from 1 to 256 bytes (8 to 2048 bits) is used to initialize a 256-byte state vector S, with elements S[0], S[1], . . ., S[255].At all times, S contains a permutation of all 8-bit numbers from 0 through 255. For encryption and decryption, a byte *k* (see Figure 2.8) is generated from S by selecting one of the 255 entries in a systematic fashion. As each value of *k* is generated, the entries in S are once again permuted.

**Table 2.3** Speed Comparisons of Symmetric Ciphers on a Pentium II **Cipher Key Length Speed (Mbps)**

DES 56 9

3DES 168 3

RC2 Variable 0.9

RC4 Variable 45

**48 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

***INITIALIZATION OF S*** To begin, the entries of S are set equal to the values from 0 == =

through 255 in ascending order; that is, S[0] 0, S[1] 1, . . ., S[255] 255. A temporary vector, T, is also created. If the length of the key K is 256 bytes, then K is transferred to T. Otherwise, for a key of length *keylen* bytes, the first *keylen* elements of T are copied from K, and then K is repeated as many times as necessary to fill out T. These preliminary operations can be summarized as:

/\* **Initialization** \*/

=

**for** i 0 **to** 255 **do**

=

S[i] i;

=

T[i] K[i **mod** keylen];

Next we use T to produce the initial permutation of S. This involves starting with S[0] and going through to S[255] and, for each S[i], swapping S[i] with another byte in S according to a scheme dictated by T[i]:

/\* **Initial Permutation of S** \*/

=

j 0;

=

**for** i 0 **to** 255 **do**

++=

j (j S[i] T[i]) **mod** 256;

Swap (S[i], S[j]);

Because the only operation on S is a swap, the only effect is a permutation. S still contains all the numbers from 0 through 255.

***STREAM GENERATION*** Once the S vector is initialized, the input key is no longer used. Stream generation involves cycling through all the elements of S[i] and, for each S[i], swapping S[i] with another byte in S according to a scheme dictated by the current configuration of S. After S[255] is reached, the process continues, starting over again at S[0]:

/\* **Stream Generation** \*/

=

i, j 0;

**while** (true)

+=

i (i 1) **mod** 256;

+=

j (j S[i]) **mod** 256;

**Swap** (S[i], S[j]);

= +

t (S[i] S[j]) **mod** 256;

=

k S[t];

To encrypt, XOR the value *k* with the next byte of plaintext. To decrypt, XOR the value *k* with the next byte of ciphertext.

Figure 2.9 illustrates the RC4 logic.

***STRENGTH OF RC4*** A number of papers have been published analyzing methods of attacking RC4 (e.g., [KNUD98], [MIST98], [FLUH00], [MANT01], [PUDO02], [PAUL03], [PAUL04]). None of these approaches is practical against RC4 with a reasonable key length, such as 128 bits. A more serious problem is reported in

**4 3**

**2**

**1**

**0 S**

**n**

**e**

**l**

**ye**

**k**

**T**

**K**

**T**

**d**

**n**

**a**

**S**

**f**

**o**

**e**

**t**

**a**

**t**

**s**

**l**

**a**

**i**

**t**

**i**

**n**

**I**

**)**

**a**

**(**

**] i**

**[T**

**T**

**] i**

**[**

**T**

**]i**

**[**

**S**

**j**

**j**

**] j[**

**S**

**] i**

**[S**

**S**

**p**

**a**

**w**

**S**

**i**

**S**

**f**

**o**

**n**

**o**

**it**

**a**

**t**

**u**

**m**

**r**

**e**

**p**

**l**

**a**

**i**

**t**

**i**

**n**

**I**

**)**

**b(**

**] i**

**[**

**S**

**j**

**j**

**] t**

**[ S**

**]j[**

**S**

**]i**

**[S**

**S**

**p a**

**w**

**Si**

**]**

**j[**

**S**

**]i**

**[**

**S**

**t**

**k**

**n**

**o**

**it**

**a**

**r**

**e**

**n**

**e**

**g**

**m**

**a**

**e**

**r**

**t**

**S**

**)**

**c**

**(**

4 C

R

**9 .**

**2**

**e**

**r**

**u**

**gi**

**F**

**49**

**50 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

[FLUH01]. The authors demonstrate that the WEP protocol, intended to provide confidentiality on 802.11 wireless LAN networks, is vulnerable to a particular attack approach. In essence, the problem is not with RC4 itself but the way in which keys are generated for use as input to RC4.This particular problem does not appear to be relevant to other applications using RC4 and can be remedied in WEP by changing the way in which keys are generated. This problem points out the difficulty in designing a secure system that involves both cryptographic functions and protocols that make use of them.

**2.5 CIPHER BLOCK MODES OF OPERATION**

A symmetric block cipher processes one block of data at a time. In the case of DES and 3DES, the block length is *b* 64 bits; for AES, the block length is *b* 128 bits. For

= =

longer amounts of plaintext, it is necessary to break the plaintext into *b*-bit blocks (padding the last block if necessary). To apply a block cipher in a variety of applica tions, five **modes of operation** have been defined by NIST (Special Publication 800- 38A). The five modes are intended to cover virtually all of the possible applications of encryption for which a block cipher could be used. These modes are intended for use with any symmetric block cipher, including triple DES and AES. The most important modes are described briefly in the remainder of this section.

**Electronic Codebook Mode**

The simplest way to proceed is using what is known as **electronic codebook (ECB) mode**, in which plaintext is handled *b* bits at a time and each block of plaintext is encrypted using the same key. The term *codebook* is used because, for a given key, there is a unique ciphertext for every *b*-bit block of plaintext. Therefore, one can imagine a gigantic codebook in which there is an entry for every possible *b*-bit plain text pattern showing its corresponding ciphertext.

With ECB, if the same *b*-bit block of plaintext appears more than once in the message, it always produces the same ciphertext. Because of this, for lengthy mes sages, the ECB mode may not be secure. If the message is highly structured, it may be possible for a cryptanalyst to exploit these regularities. For example, if it is known that the message always starts out with certain predefined fields, then the cryptana lyst may have a number of known plaintext–ciphertext pairs to work with. If the message has repetitive elements with a period of repetition a multiple of *b* bits, then these elements can be identified by the analyst.This may help in the analysis or may provide an opportunity for substituting or rearranging blocks.

To overcome the security deficiencies of ECB, we would like a technique in which the same plaintext block, if repeated, produces different ciphertext blocks.

**Cipher Block Chaining Mode**

In the **cipher block chaining (CBC) mode** (Figure 2.10), the input to the encryption algorithm is the XOR of the current plaintext block and the preceding ciphertext block; the same key is used for each block. In effect, we have chained together the processing of the sequence of plaintext blocks. The input to the encryption function

**2.5 / CIPHER BLOCK MODES OF OPERATION 51**

***IV K***

***K***

***IV***

***P*1**

**Encrypt *C*1**

***C*1**

**Decrypt *P*1**

***K K***

***P*2**

**Encrypt**

***C*2**

**(a) Encryption *C*2**

**Decrypt**

***P*2**

**(b) Decryption**

***CN*–1 *K***

***K***

**C*N*–1**

***PN***

**Encrypt *CN***

***CN***

**Decrypt *PN***

**Figure 2.10** Cipher Block Chaining (CBC) Mode

for each plaintext block bears no fixed relationship to the plaintext block. Therefore, repeating patterns of *b* bits are not exposed.

For decryption, each cipher block is passed through the decryption algorithm. The result is XORed with the preceding ciphertext block to produce the plaintext block. To see that this works, we can write

*Cj* E(*K*, [*Cj* 1 *Pj*])

= -

where E[*K*, *X*] is the encryption of plaintext *X* using key *K*, and is the exclusive

OR operation.Then

D(*K*, *Cj*) D(*K*, E(*K*, [*Cj* 1 *Pj*]))

= -

= -

D(*K*, *Cj*) *Cj* 1 *Pj*

-  = -  -  =

*Cj* 1 D(*K*, *Cj*) *Cj* 1 *Cj* 1 *Pj Pj*

which verifies Figure 2.10b.

To produce the first block of ciphertext, an initialization vector (IV) is XORed with the first block of plaintext. On decryption, the IV is XORed with the output of the decryption algorithm to recover the first block of plaintext.

The IV must be known to both the sender and receiver. For maximum security, the IV should be protected as well as the key. This could be done by sending the IV

**52 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

using ECB encryption. One reason for protecting the IV is as follows: If an opponent is able to fool the receiver into using a different value for IV, then the opponent is able to invert selected bits in the first block of plaintext.To see this, consider the following:

*C*1 E(*K*, [IV *P*1])

=

*P*1 IV D(*K*, *C*1)

=

Now use the notation that *X*[*j*] denotes the *j*th bit of the *b*-bit quantity *X*. Then *P*1[*i*] IV[*i*] D(*K*, *C*1)[*i*]

=

Then, using the properties of XOR, we can state

*P*1[*i*]' IV[*i*]' D(*K*, *C*1)[*i*]

=

where the prime notation denotes bit complementation. This means that if an oppo nent can predictably change bits in IV, the corresponding bits of the received value of *P*1 can be changed.

**Cipher Feedback Mode**

It is possible to convert any block cipher into a stream cipher by using the **cipher feedback (CFB) mode**. A stream cipher eliminates the need to pad a message to be an integral number of blocks. It also can operate in real time. Thus, if a character stream is being transmitted, each character can be encrypted and transmitted imme diately using a character-oriented stream cipher.

One desirable property of a stream cipher is that the ciphertext be of the same length as the plaintext.Thus, if 8-bit characters are being transmitted, each character should be encrypted using 8 bits. If more than 8 bits are used, transmission capacity is wasted.

Figure 2.11 depicts the CFB scheme. In the figure, it is assumed that the unit of =

transmission is *s* bits; a common value is *s* 8. As with CBC, the units of plaintext are chained together, so that the ciphertext of any plaintext unit is a function of all the preceding plaintext.

First, consider encryption. The input to the encryption function is a *b*-bit shift register that is initially set to some initialization vector (IV). The leftmost (most significant) *s* bits of the output of the encryption function are XORed with the first unit of plaintext *P*1 to produce the first unit of ciphertext *C*1, which is then transmitted. In addition, the contents of the shift register are shifted left by *s* bits, and *C*1 is placed in the rightmost (least significant) *s* bits of the shift register. This process continues until all plaintext units have been encrypted.

For decryption, the same scheme is used, except that the received ciphertext unit is XORed with the output of the encryption function to produce the plaintext unit. Note that it is the *encryption* function that is used, not the decryption function. This is easily explained. Let S*s*(*X*) be defined as the most significant *s* bits of *X*. Then *C*1 *P*1 S*s*[E(*K*, IV)]

=

Therefore,

*P*1 *C*1 S*s*[E(*K*, IV)]

=

The same reasoning holds for subsequent steps in the process.

**2.5 / CIPHER BLOCK MODES OF OPERATION 53 *CN*–1**

***K***

***s* bits**

***P*1**

**IV**

**Encrypt**

***s* bits Discard**

**Select**

***b* – *s* bits**

***C*1**

***K***

***s* bits**

***P*2**

***b – s* bits *s* bits Shift register Encrypt**

**Select**

***s* bitsDiscard *b* – *s* bits**

***C*2**

***K***

***s* bits**

***PN***

***b – s* bits *s* bits Shift register Encrypt**

**Select**

***s* bitsDiscard *b* – *s* bits**

***CN***

***s* bits *s* bits**

**(a) Encryption**

***CN*–1**

***s* bits**

**IV**

***K***

***b – s* bits *s* bits Shift register *b – s* bits *s* bits Shift register *K***

***K***

**Encrypt**

***s* bits Discard**

**Select**

***b* – *s* bits**

***P*1**

***C*1**

***s* bits**

**Encrypt**

**Select**

***s* bitsDiscard *b* – *s* bits**

***P*2**

***C*2**

***s* bits**

**Encrypt**

**Select**

***s* bitsDiscard *b* – *s* bits**

***CN***

***s* bits**

***PN***

***s* bits *s* bits**

**(b) Decryption**

**Figure 2.11** *s*-bit Cipher Feedback (CFB) Mode**Counter Mode**

***s* bits**

Although interest in the **counter mode (CTR)** has increased recently, with applica tions to ATM (asynchronous transfer mode) network security and IPSec (IP secu rity), this mode was proposed early on (e.g., [DIFF79]).

Figure 2.12 depicts the CTR mode. A counter equal to the plaintext block size is used.The only requirement stated in SP 800-38A is that the counter value must be different for each plaintext block that is encrypted. Typically, the counter is initial ized to some value and then incremented by 1 for each subsequent block (modulo

**54 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

**Counter 1**

***K***

**Counter 2 Counter *N***

***K***

***K***

***P*1**

**Encrypt *C*1**

**Encrypt**

***P*2 *PN***

***C*2**

**(a) Encryption**

**Encrypt *CN***

**Counter 1**

***K***

**Counter 2 Counter *N***

***K***

***K***

***C*1**

**Encrypt *P*1**

**Encrypt**

***C*2 *CN***

***P*2**

**(b) Decryption**

**Encrypt *PN***

**Figure 2.12** Counter (CTR) Mode

2*b*, where *b* is the block size). For encryption, the counter is encrypted and then XORed with the plaintext block to produce the ciphertext block; there is no chain ing. For decryption, the same sequence of counter values is used, with each encrypted counter XORed with a ciphertext block to recover the corresponding plaintext block.

[LIPM00] lists the following advantages of CTR mode.

• **Hardware efficiency:** Unlike the chaining modes, encryption (or decryption) in CTR mode can be done in parallel on multiple blocks of plaintext or ciphertext. For the chaining modes, the algorithm must complete the computation on one block before beginning on the next block.This limits the maximum throughput of

**2.6 / RECOMMENDED READING AND WEB SITES 55**

the algorithm to the reciprocal of the time for one execution of block encryption or decryption. In CTR mode, the throughput is only limited by the amount of parallelism that is achieved.

• **Software efficiency:** Similarly, because of the opportunities for parallel exe cution in CTR mode, processors that support parallel features (such as aggressive pipelining, multiple instruction dispatch per clock cycle, a large number of registers, and SIMD instructions) can be effectively utilized.

• **Preprocessing:** The execution of the underlying encryption algorithm does not depend on input of the plaintext or ciphertext. Therefore, if sufficient memory is available and security is maintained, preprocessing can be used to prepare the output of the encryption boxes that feed into the XOR functions in Figure 2.12. When the plaintext or ciphertext input is presented, then the only computation is a series of XORs. Such a strategy greatly enhances throughput.

• **Random access:** The *i*th block of plaintext or ciphertext can be processed in random-access fashion. With the chaining modes, block *Ci* cannot be com puted until the *i* 1 prior block are computed. There may be applications in

-

which a ciphertext is stored, and it is desired to decrypt just one block; for such applications, the random access feature is attractive.

• **Provable security:** It can be shown that CTR is at least as secure as the other modes discussed in this section.

• **Simplicity:** Unlike ECB and CBC modes, CTR mode requires only the imple mentation of the encryption algorithm and not the decryption algorithm. This matters most when the decryption algorithm differs substantially from the encryption algorithm, as it does for AES. In addition, the decryption key scheduling need not be implemented.

**2.6 RECOMMENDED READING AND WEB SITES**

The topics in this chapter are covered in greater detail in [STAL11]. For coverage of crypto graphic algorithms, [SCHN96] is an essential reference work; it contains descriptions of virtu ally every cryptographic algorithm and protocol published up to the time of the writing of the book. Another worthwhile and detailed survey is [MENE97]. A more in-depth treatment, with rigorous mathematical discussion, is [STIN06].

**MENE97** Menezes, A.; van Oorschot, P.; and Vanstone, S. *Handbook of Applied Cryptography.* Boca Raton, FL: CRC Press, 1997.

**SCHN96** Schneier, B. *Applied Cryptography.* New York: Wiley, 1996.

**STAL11** Stallings, W. *Cryptography and Network Security: Principles and Practice, Fifth Edition.* Upper Saddle River, NJ: Prentice Hall, 2011.

**STIN06** Stinson, D. *Cryptography:Theory and Practice.* Boca Raton, FL: Chapman&Hall/ CRC Press, 2006.

**56 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

**Recommended Web Sites:**

• **AES home page:** NIST’s page on AES. Contains the standard plus a number of other relevant documents.

• **AES Lounge:** Contains a comprehensive bibliography of documents and papers on AES with access to electronic copies.

• **Block Cipher Modes of Operation:** NIST page with full information on NIST-approved modes of operation.

**2.7 KEY TERMS, REVIEW QUESTIONS,AND PROBLEMS Key Terms**

Advanced Encryption

Standard (AES)

block cipher

brute-force attack

cipher block chaining (CBC) mode

cipher feedback (CFB) mode ciphertext

counter mode (CTR)

cryptanalysis

**Review Questions**

Cryptography

Data Encryption Standard (DES)

decryption

electronic codebook (ECB) mode

encryption

end-to-end encryption Feistel cipher

key distribution

keystream

link encryption

plaintext

session key

stream cipher

subkey

symmetric encryption triple DES (3DES)

**2.1** What are the essential ingredients of a symmetric cipher?

**2.2** What are the two basic functions used in encryption algorithms? **2.3** How many keys are required for two people to communicate via a symmetric cipher?

**2.4** What is the difference between a block cipher and a stream cipher? **2.5** What are the two general approaches to attacking a cipher?

**2.6** Why do some block cipher modes of operation only use encryption while others use both encryption and decryption?

**2.7** What is triple encryption?

**2.8** Why is the middle portion of 3DES a decryption rather than an encryption?

**Problems**

**2.1** This problem uses a real-world example of a symmetric cipher, from an old U.S. Special Forces manual (public domain). The document, filename *SpecialForces.pdf*, is available at this book’s Web site.

**2.7 / KEY TERMS, REVIEW QUESTIONS,AND PROBLEMS 57**

**a.** Using the two keys (memory words) *cryptographic* and *network security*, encrypt the following message:

Be at the third pillar from the left outside the lyceum theatre tonight at seven. If you are distrustful bring two friends.

Make reasonable assumptions about how to treat redundant letters and excess letters in the memory words and how to treat spaces and punctuation. Indicate what your assumptions are. *Note:* The message is from the Sherlock Holmes novel, *The Sign of Four*.

**b.** Decrypt the ciphertext. Show your work.

**c.** Comment on when it would be appropriate to use this technique and what its advantages are.

**2.2** Consider a very simple symmetric block encryption algorithm in which 32-bits blocks of plaintext are encrypted using a 64-bit key. Encryption is defined as = n+

*C* (*P K*0) *K*1

= = = =

where *C* ciphertext, *K* secret key, *K*0 leftmost 64 bits of *K*, *K*1 rightmost = n+

64 bits of *K*, bitwise exclusive OR, and is addition mod 264. **a.** Show the decryption equation. That is, show the equation for *P* as a function of *C*, *K*0, and *K*1.

**b.** Suppose and adversary has access to two sets of plaintexts and their correspond ing ciphertexts and wishes to determine *K*. We have the two equations:

= n+ = n+

*C* (*P K*0) *K*1; *C*' (*P*' *K*0) *K*1

First, derive an equation in one unknown (e.g., *K*0). Is it possible to proceed fur ther to solve for *K*0? **2.3** Perhaps the simplest “serious” symmetric block encryption algorithm is the Tiny Encryption Algorithm (TEA). TEA operates on 64-bit blocks of plaintext using a 128-bit key. The plaintext is divided into two 32-bit blocks (*L*0, *R*0), and the key is divided into four 32-bit blocks (*K*0, *K*1, *K*2, *K*3). Encryption involves repeated applica tion of a pair of rounds, defined as follows for rounds *i* and *i* 1:

+

= -

*Li Ri* 1

= - n+ -

*Ri Li* 1 F(*Ri* 1, *K*0, *K*1, δ*i*)

+ =

*Li* 1 *Ri*

+ = n+ +

*Ri* 1 *Li* F(*Ri*, *K*2, *K*3, δ*i* 1)

where F is defined as

= 6 6 n+ 7 7 n+ n+

F(*M*, *Kj*, *Kk*, δ*i*) ((M 4) *Kj*) ((M 5) *Kk*) (M δ*i*) 6 6

and where the logical shift of *x* by *y* bits is denoted by *x y*, the logical right shift of *x* by *y* bits is denoted by *x y*, and δ*i* is a sequence of predetermined constants.

7 7

**a.** Comment on the significance and benefit of using the sequence of constants. **b.** Illustrate the operation of TEA using a block diagram or flow chart type of depiction.

**c.** If only one pair of rounds is used, then the ciphertext consists of the 64-bit block (*L*2, *R*2). For this case, express the decryption algorithm in terms of equations.

**d.** Repeat part (c) using an illustration similar to that used for part (b). **2.4** Show that Feistel decryption is the inverse of Feistel encryption. **2.5** Consider a Feistel cipher composed of 16 rounds with block length 128 bits and key

length 128 bits. Suppose that, for a given *k*, the key scheduling algorithm determines values for the first eight round keys, *k*1, *k*2, . . ., *k*8, and then sets

*k*9 *k*8, *k*10 *k*7, *k*11 *k*6 === , . . ., *k*16 = *k*1

**58 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

Suppose you have a ciphertext *c*. Explain how, with access to an encryption oracle, you can decrypt *c* and determine *m* using just a single oracle query. This shows that such a cipher is vulnerable to a chosen plaintext attack. (An encryption oracle can be thought of as a device that, when given a plaintext, returns the corresponding ciphertext. The internal details of the device are not known to you, and you cannot break open the device. You can only gain information from the oracle by making queries to it and observing its responses.)

**2.6** For any block cipher, the fact that it is a nonlinear function is crucial to its security. To see this, suppose that we have a linear block cipher EL that encrypts 128-bit blocks of plaintext into 128-bit blocks of ciphertext. Let EL(*k*, *m*) denote the encryption of a 128-bit message *m* under a key *k* (the actual bit length of *k* is irrel evant). Thus,

=

EL(*k*, [*m*1 *m*2]) EL(*k*, *m*1) EL(*k*, *m*2) for all 128-bit patterns *m*1, *m*2

Describe how, with 128 chosen ciphertexts, an adversary can decrypt any ciphertext without knowledge of the secret key *k*. (A “chosen ciphertext” means that an adver sary has the ability to choose a ciphertext and then obtain its decryption. Here, you have 128 plaintext–ciphertext pairs to work with, and you have the ability to chose the value of the ciphertexts.)

**2.7** Suppose you have a true random bit generator where each bit in the generated stream has the same probability of being a 0 or 1 as any other bit in the stream and that the bits are not correlated; that is, the bits are generated from identical indepen +

dent distribution. However, the bit stream is biased. The probability of a 1 is 0.5 δ - 66

and the probability of a 0 is 0.5 δ, where 0 δ 0.5.A simple deskewing algorithm is as follows: Examine the bit stream as a sequence of non-overlapping pairs. Discard all 00 and 11 pairs. Replace each 01 pair with 0 and each 10 pair with 1.

**a.** What is the probability of occurrence of each pair in the original sequence? **b.** What is the probability of occurrence of 0 and 1 in the modified sequence? **c.** What is the expected number of input bits to produce *x* output bits?

**d.** Suppose that the algorithm uses overlapping successive bit pairs instead of nonoverlapping successive bit pairs. That is, the first output bit is based on input bits 1 and 2, the second output bit is based on input bits 2 and 3, and so on. What can you say about the output bit stream?

**2.8** Another approach to deskewing is to consider the bit stream as a sequence of non-overlapping groups of *n* bits each and output the parity of each group. That is, if a group contains an odd number of ones, the output is 1; otherwise the output is 0.

**a.** Express this operation in terms of a basic Boolean function.

**b.** Assume, as in the Problem 2.7, that the probability of a 1 is 0.5 δ. If each group

+

consists of 2 bits, what is the probability of an output of 1?

**c.** If each group consists of 4 bits, what is the probability of an output of 1?

**d.** Generalize the result to find the probability of an output of 1 for input groups of *n* bits.

**2.9** What RC4 key value will leave S unchanged during initialization? That is, after the initial permutation of S, the entries of S will be equal to the values from 0 through 255 in ascending order.

**2.10** RC4 has a secret internal state which is a permutation of all the possible values of the vector **S** and the two indices *i* and *j*.

**a.** Using a straightforward scheme to store the internal state, how many bits are used?

**b.** Suppose we think of it from the point of view of how much information is repre sented by the state. In that case, we need to determine how may different states

**2.7 / KEY TERMS, REVIEW QUESTIONS,AND PROBLEMS 59**

there are, then take the log to the base 2 to find out how many bits of information this represents. Using this approach, how many bits would be needed to represent the state?

**2.11** Alice and Bob agree to communicate privately via e-mail using a scheme based on RC4, but they want to avoid using a new secret key for each transmission. Alice and Bob privately agree on a 128-bit key *k*. To encrypt a message *m* consisting of a string of bits, the following procedure is used.

**1.** Choose a random 80-bit value *v*

=

**2.** Generate the ciphertext *c* RC4(*v* 7 *k*) *m*

**3.** Send the bit string (*v* 7 *c*)

**a.** Suppose Alice uses this procedure to send a message *m* to Bob. Describe how Bob can recover the message *m* from (*v* 7 *c*) using *k*.

**b.** If an adversary observes several values (*v*1 7 *c*1), (*v*2 7 *c*2), . . . transmitted between Alice and Bob, how can he/she determine when the same key stream has been used to encrypt two messages?

**2.12** With the ECB mode, if there is an error in a block of the transmitted ciphertext, only the corresponding plaintext block is affected. However, in the CBC mode, this error propagates. For example, an error in the transmitted *C*1 (Figure 2.10) obviously cor rupts *P*1 and *P*2. **a.** Are any blocks beyond *P*2 affected?

**b.** Suppose that there is a bit error in the source version of *P*1. Through how many ciphertext blocks is this error propagated? What is the effect at the receiver?

**2.13** Is it possible to perform encryption operations in parallel on multiple blocks of plain text in CBC mode? How about decryption?

**2.14** Suppose an error occurs in a block of ciphertext on transmission using CBC. What effect is produced on the recovered plaintext blocks?

**2.15** CBC-Pad is a block cipher mode of operation used in the RC5 block cipher, but it could be used in any block cipher. CBC-Pad handles plaintext of any length. The ciphertext is longer than the plaintext by at most the size of a single block. Padding is used to assure that the plaintext input is a multiple of the block length. It is assumed that the original plaintext is an integer number of bytes. This plain text is padded at the end by from 1 to *bb* bytes, where *bb* equals the block size in bytes. The pad bytes are all the same and set to a byte that represents the number of bytes of padding. For example, if there are 8 bytes of padding, each byte has the bit pattern **00001000**. Why not allow zero bytes of padding? That is, if the orig inal plaintext is an integer multiple of the block size, why not refrain from padding?

**2.16** Padding may not always be appropriate. For example, one might wish to store the encrypted data in the same memory buffer that originally contained the plaintext. In that case, the ciphertext must be the same length as the original plaintext. A mode for that purpose is the ciphertext stealing (CTS) mode. Figure 2.13a shows an implemen tation of this mode.

**a.** Explain how it works.

**b.** Describe how to decrypt C*n* 1 and C*n*.

-

**2.17** Figure 2.13b shows an alternative to CTS for producing ciphertext of equal length to the plaintext when the plaintext is not an integer multiple of the block size. **a.** Explain the algorithm.

**b.** Explain why CTS is preferable to this approach illustrated in Figure 2.13b. **2.18** If a bit error occurs in the transmission of a ciphertext character in 8-bit CFB mode, how far does the error propagate?

**60 CHAPTER 2 / SYMMETRIC ENCRYPTION AND MESSAGE CONFIDENTIALITY**

**IV *P*1**

***PN*****2**

***PN*****1**

***PN* 00…0**

*C****N*****3**

***K K K K***

**Encrypt Encrypt**

**Encrypt**

**Encrypt**

***C*1**

***CN*****2**

***CN* X**

***CN*****1**

**(a) Cipheretext stealing mode**

**IV**

***P*1**

**(*bb* bits)**

***CN*****3**

***PN*****2**

**(*bb* bits)**

***PN*****1**

**(*bb* bits)**

***PN***

**( *j* bits)**

**Select**

***K K K K* Encrypt Encrypt Encrypt Encrypt**

**leftmost j bits**

***C*1**

**(*bb* bits)**

***CN*****2**

**(*bb* bits)**

***CN*****1**

**(*bb* bits)**

***CN***

**( *j* bits)**

**(b) Alternative method**

**Figure 2.13** Block Cipher Modes for Plaintext not a Multiple of Block Size

**CHAPTER**

**PUBLIC-KEY CRYPTOGRAPHY AND MESSAGE AUTHENTICATION**

**3.1 Approaches to Message Authentication**

Authentication Using Conventional Encryption

Message Authentication without Message Encryption

**3.2 Secure Hash Functions**

Hash Function Requirements

Security of Hash Functions

Simple Hash Functions

The SHA Secure Hash Function

**3.3 Message Authentication Codes**

HMAC

MACs Based on Block Ciphers

**3.4 Public-Key Cryptography Principles**

Public-Key Encryption Structure

Applications for Public-Key Cryptosystems

Requirements for Public-Key Cryptography

**3.5 Public-Key Cryptography Algorithms**

The RSA Public-Key Encryption Algorithm

Diffie-Hellman Key Exchange

Other Public-Key Cryptography Algorithms

**3.6 Digital Signatures**

**3.7 Recommended Reading and Web Sites**

**3.8 Key Terms, Review Questions, and Problems**

**61**

**62 CHAPTER 3 / PUBLIC-KEY CRYPTOGRAPHY AND MESSAGE AUTHENTICATION**

*Every Egyptian received two names, which were known respectively as the true name and the good name, or the great name and the little name; and while the good or little name was made public, the true or great name appears to have been carefully concealed.*

—*The Golden Bough,* Sir James George Frazer

*To guard against the baneful influence exerted by strangers is therefore an ele mentary dictate of savage prudence. Hence before strangers are allowed to enter a district, or at least before they are permitted to mingle freely with the inhabitants, certain ceremonies are often performed by the natives of the country for the pur pose of disarming the strangers of their magical powers, or of disinfecting, so to speak, the tainted atmosphere by which they are supposed to be surrounded.*

—*The Golden Bough*, Sir James George Frazer

In addition to message confidentiality, message authentication is an important network security function. This chapter examines three aspects of message authentication. First, we look at the use of message authentication codes and hash functions to provide message authentication. Then we look at public-key encryp tion principles and two specific public-key algorithms. These algorithms are useful in the exchange of conventional encryption keys. Then we look at the use of public-key encryption to produce digital signatures, which provides an enhanced form of message authentication.

**3.1 APPROACHES TO MESSAGE AUTHENTICATION**

Encryption protects against passive attack (eavesdropping).A different requirement is to protect against active attack (falsification of data and transactions). Protection against such attacks is known as message authentication.

A message, file, document, or other collection of data is said to be authentic when it is genuine and comes from its alleged source. Message authentication is a procedure that allows communicating parties to verify that received messages are authentic.1 The two important aspects are to verify that the contents of the message have not been altered and that the source is authentic. We may also wish to verify a message’s timeliness (it has not been artificially delayed and replayed) and sequence relative to other messages flowing between two parties. All of these concerns come under the category of data integrity as described in Chapter 1.

**Authentication Using Conventional Encryption**

It would seem possible to perform authentication simply by the use of symmetric encryption. If we assume that only the sender and receiver share a key (which is as it should be), then only the genuine sender would be able to encrypt a message

1For simplicity, for the remainder of this chapter, we refer to *message authentication.* By this we mean both authentication of transmitted messages and of stored data (*data authentication*).

**3.1 / APPROACHES TO MESSAGE AUTHENTICATION 63**

successfully for the other participant, provided the receiver can recognize a valid message. Furthermore, if the message includes an error-detection code and a sequence number, the receiver is assured that no alterations have been made and that sequencing is proper. If the message also includes a timestamp, the receiver is assured that the message has not been delayed beyond that normally expected for network transit.

In fact, symmetric encryption alone is not a suitable tool for data authentica tion. To give one simple example, in the ECB mode of encryption, if an attacker reorders the blocks of ciphertext, then each block will still decrypt successfully. However, the reordering may alter the meaning of the overall data sequence. Although sequence numbers may be used at some level (e.g., each IP packet), it is typically not the case that a separate sequence number will be associated with each *b*-bit block of plaintext. Thus, block reordering is a threat.

**Message Authentication without Message Encryption**

In this section, we examine several approaches to message authentication that do not rely on encryption. In all of these approaches, an authentication tag is generated and appended to each message for transmission. The message itself is not encrypted and can be read at the destination independent of the authentication function at the destination.

Because the approaches discussed in this section do not encrypt the message, message confidentiality is not provided. As was mentioned, message encryption by itself does not provide a secure form of authentication. However, it is possible to combine authentication and confidentiality in a single algorithm by encrypting a message plus its authentication tag. Typically, however, message authentication is provided as a separate function from message encryption. [DAVI89] suggests three situations in which message authentication without confidentiality is preferable:

**1.** There are a number of applications in which the same message is broadcast to a number of destinations. Two examples are notification to users that the net work is now unavailable and an alarm signal in a control center. It is cheaper and more reliable to have only one destination responsible for monitoring authenticity. Thus, the message must be broadcast in plaintext with an associ ated message authentication tag. The responsible system performs authenti cation. If a violation occurs, the other destination systems are alerted by a general alarm.

**2.** Another possible scenario is an exchange in which one side has a heavy load and cannot afford the time to decrypt all incoming messages. Authentication is car ried out on a selective basis with messages being chosen at random for checking.

**3.** Authentication of a computer program in plaintext is an attractive service.The computer program can be executed without having to decrypt it every time, which would be wasteful of processor resources. However, if a message authentication tag were attached to the program, it could be checked when ever assurance is required of the integrity of the program.

Thus, there is a place for both authentication and encryption in meeting security requirements.

**64 CHAPTER 3 / PUBLIC-KEY CRYPTOGRAPHY AND MESSAGE AUTHENTICATION**

***MESSAGE AUTHENTICATION CODE*** One authentication technique involves the use of a secret key to generate a small block of data, known as a **message authentication code (MAC)**, that is appended to the message. This technique assumes that two communicating parties, say A and B, share a common secret key *KAB.* When A has a message to send to B, it calculates the message authentication code as a function of the message and the key: MAC*M* F(*KAB*, *M*).The message plus code are transmitted to

=

the intended recipient. The recipient performs the same calculation on the received message, using the same secret key, to generate a new message authentication code. The received code is compared to the calculated code (Figure 3.1). If we assume that only the receiver and the sender know the identity of the secret key, and if the received code matches the calculated code, then the following statements apply:

**1.** The receiver is assured that the message has not been altered. If an attacker alters the message but does not alter the code, then the receiver’s calculation of the code will differ from the received code. Because the attacker is assumed not to know the secret key, the attacker cannot alter the code to correspond to the alterations in the message.

**2.** The receiver is assured that the message is from the alleged sender.Because no one else knows the secret key, no one else could prepare a message with a proper code. **3.** If the message includes a sequence number (such as is used with HDLC and TCP), then the receiver can be assured of the proper sequence, because an attacker cannot successfully alter the sequence number.

Message

*K*

MAC

algorithm *K*

MAC

Transmit

MAC

algorithm Compare

**Figure 3.1** Message Authentication Using a Message Authentication Code (MAC)

**3.1 / APPROACHES TO MESSAGE AUTHENTICATION 65**

A number of algorithms could be used to generate the code.The NIST specifi cation, FIPS PUB 113, recommends the use of DES. DES is used to generate an encrypted version of the message, and the last number of bits of ciphertext are used as the code. A 16- or 32-bit code is typical.

The process just described is similar to encryption. One difference is that the authentication algorithm need not be reversible, as it must for decryption. Because of the mathematical properties of the authentication function, it is less vulnerable to being broken than encryption.

***ONE-WAY HASH FUNCTION*** An alternative to the message authentication code is the **one-way hash function**. As with the message authentication code, a hash function accepts a variable-size message *M* as input and produces a fixed-size message digest H(*M*) as output. Unlike the MAC, a hash function does not take a secret key as input. To authenticate a message, the message digest is sent with the message in such a way that the message digest is authentic.

Figure 3.2 illustrates three ways in which the message can be authenticated. The message digest can be encrypted using conventional encryption (part a); if it is assumed that only the sender and receiver share the encryption key, then authentic ity is assured. The message digest can be encrypted using public-key encryption (part b); this is explained in Section 3.5. The public-key approach has two advan tages: (1) It provides a digital signature as well as message authentication. (2) It does not require the distribution of keys to communicating parties.

These two approaches also have an advantage over approaches that encrypt the entire message in that less computation is required. Nevertheless, there has been interest in developing a technique that avoids encryption altogether. Several reasons for this interest are pointed out in [TSUD92]:

• Encryption software is quite slow. Even though the amount of data to be encrypted per message is small, there may be a steady stream of messages into and out of a system.

• Encryption hardware costs are nonnegligible. Low-cost chip implementations of DES are available, but the cost adds up if all nodes in a network must have this capability.

• Encryption hardware is optimized toward large data sizes. For small blocks of data, a high proportion of the time is spent in initialization/invocation overhead. • An encryption algorithm may be protected by a patent.

Figure 3.2c shows a technique that uses a hash function but no encryption for message authentication. This technique assumes that two communicating parties, say A and B, share a common secret value *SAB.* When A has a message to send to B, it calculates the hash function over the concatenation of the secret value and the message: *MDM* H(*SAB*7*M*).2 It then sends [*M*7*MDM*] to B. Because B possesses

=

*SAB*, it can recompute H(*SAB*7*M*) and verify *MDM.* Because the secret value itself is 2 7 denotes concatenation.