CRYPTOGRAPHY AND DATA SECURITY

DOROTHY ELIZABETH ROBLING DENNING

VMware Ex. 1016
VMware, LLC, v. Lionra Technologies Ltd.
IPR2024-00302 Page 1 of 18

Cryptography and Data Security

Dorothy Elizabeth Robling Denning
PURDUE UNIVERSITY

ADDISON-WESLEY PUBLISHING COMPANY
Reading, Massachusetts ■ Menlo Park, California
London ■ Amsterdam ■ Don Mills, Ontario ■ Sydney

Library of Congress Cataloging in Publication Data

Denning, Dorothy E., (Dorothy Elizabeth), 1945–
Cryptography and data security.

Includes bibliographical references and index.
1. Computers—Access control. 2. Cryptography.
3. Data protection. I. Title.
QA76.9.A25D46 1982 001.64'028'9 81–15012
ISBN 0–201–10150–5 AACR2

Copyright © 1982 by Addison-Wesley Publishing Company, Inc.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher. Printed in the United States of America. Published simultaneously in Canada.

ISBN 0-201-10150-5
ABCDEFGHIJ-MA-898765432

In memory of my Father,

Cornelius Lowell Robling

1910–1965

Preface

Electronic computers have evolved from exiguous experimental enterprises in the 1940s to prolific practical data processing systems in the 1980s. As we have come to rely on these systems to process and store data, we have also come to wonder about their ability to protect valuable data.

Data security is the science and study of methods of protecting data in computer and communication systems from unauthorized disclosure and modification. The goal of this book is to introduce the mathematical principles of data security and to show how these principles apply to operating systems, database systems, and computer networks. The book is for students and professionals seeking an introduction to these principles. There are many references for those who would like to study specific topics further.

Data security has evolved rapidly since 1975. We have seen exciting developments in cryptography: public-key encryption, digital signatures, the Data Encryption Standard (DES), key safeguarding schemes, and key distribution protocols. We have developed techniques for verifying that programs do not leak confidential data, or transmit classified data to users with lower security clearances. We have found new controls for protecting data in statistical databases—and new methods of attacking these databases. We have come to a better understanding of the theoretical and practical limitations to security.

Because the field is evolving so rapidly, it has been difficult to write a book that is both coherent and current. Even as the manuscript was in production, there were new developments in the field. Although I was able to incorporate a few of these developments, they are not as well integrated into the book as I would like. In many cases, I was only able to include references.

Some areas are still unsettled, and I was unable to treat them to my satisfaction. One such area is operating system verification; another is the integration of

VMware Ex. 1016
Page 6 of 18

cryptographic controls into operating systems and database systems. I hope to cover these topics better in later editions of the book.

Data security draws heavily from mathematics and computer science. I have assumed my audience has some background in programming, data structures, operating systems, database systems, computer architecture, probability theory, and linear algebra. Because I have found most computer science students have little background in information theory and number theory, I have included self-contained tutorials on these subjects. Because complexity theory is a relatively new area, I have also summarized it.

This book is used in a one-semester graduate computer science course at Purdue University. The students are assigned exercises, programming projects, and a term project. The book is suitable for a graduate or advanced undergraduate course and for independent study. There are a few exercises at the end of each chapter, most of which are designed so the reader can recognize the right answer. I have purposely not included solutions. There is also a puzzle.

Here is a brief summary of the chapters:

- *Chapter 1, Introduction,* introduces the basic concepts of cryptography, data security, information theory, complexity theory, and number theory.
- *Chapter 2, Encryption Algorithms,* describes both classical and modern encryption algorithms, including the Data Encryption Standard (DES) and public-key algorithms.
- *Chapter 3, Cryptographic Techniques,* studies various techniques related to integrating cryptographic controls into computer systems, including key management.
- *Chapter 4, Access Controls,* describes the basic principles of mechanisms that control access by subjects (e.g., users or programs) to objects (e.g., files and records). These mechanisms regulate direct access to objects, but not what happens to the information contained in these objects.
- *Chapter 5, Information Flow Controls,* describes controls that regulate the dissemination of information. These controls are needed to prevent programs from leaking confidential data, or from disseminating classified data to users with lower security clearances.
- *Chapter 6, Inference Controls,* describes controls that protect confidential data released as statistics about subgroups of individuals.

I am deeply grateful to Jim Anderson, Bob Blakley, Peter Denning, Whit Diffie, Peter Neumann, and Rich Reitman, whose penetrating criticisms and suggestions guided me to important results and helped me focus my ideas. I am also grateful to Greg Andrews, Leland Beck, Garrett Birkhoff, Manuel Blum, David Chaum, Francis Chin, Larry Cox, Töre Dalenius, George Davida, Dave Gifford, Carl Hammer, Mike Harrison, Chris Hoffmann, Stephen Matyas, Glen Myers, Bob Morris, Steve Reiss, Ron Rivest, Jan Schlörer, Gus Simmons, and Larry Snyder. These people gave generously of their time to help make this a better book.

VMware Ex. 1016
Page 7 of 18

I am thankful to the students who read the book, worked the problems, and provided numerous comments and suggestions: George Adams, Brian Beuning, Steve Booth, Steve Breese, Carl Burch, Steve Burton, Ray Ciesielski, Cliff Cockerham, Ken Dickman, James Dobrina, Dave Eckert, Jeremy Epstein, Tim Field, Jack Fitch, Jim Fuss, Greg Gardner, Neil Harrison, Ching-Chih Hsiao, Teemu Kerola, Ron Krol, Meng Lee, Peter Liesenfelt, Paul Morrisett, Tim Nodes, Bhasker Parthasarathy, Steve Pauley, Alan Pieramico, Steve Raiman, Dan Reed, David Rutkin, Paul Scherf, Carl Smith, Alan Stanson, Mark Stinson, Andy Tong, and Kim Tresner. I am especially thankful to Matt Bishop for providing solutions and for grading.

The working version of the book was prepared on the department's VAX computer. I am grateful to Doug Comer, Herb Schwetman, and the many others who kept the system operational and paid careful attention to backup procedures. I am grateful to the people who helped with the publication of the book, especially Peter Gordon, Gail Goodell, Cheryl Wurzbacher, and Judith Gimple.

I am especially grateful to my husband, Peter, for his encouragement, support, advice, and help throughout.

VMware Ex. 1016
Page 8 of 18

Contents

1 INTRODUCTION 1
- 1.1 Cryptography 1
- 1.2 Data Security 3
- 1.3 Cryptographic Systems 7
  - 1.3.1 Public-Key Systems 11
  - 1.3.2 Digital Signatures 14
- 1.4 Information Theory 16
  - 1.4.1 Entropy and Equivocation 17
  - 1.4.2 Perfect Secrecy 22
  - 1.4.3 Unicity Distance 25
- 1.5 Complexity Theory 30
  - 1.5.1 Algorithm Complexity 30
  - 1.5.2 Problem Complexity and **NP**-Completeness 31
  - 1.5.3 Ciphers Based on Computationally Hard Problems 34
- 1.6 Number Theory 35
  - 1.6.1 Congruences and Modular Arithmetic 36
  - 1.6.2 Computing Inverses 39
  - 1.6.3 Computing in Galois Fields 48
- Exercises 54
- References 56

2 ENCRYPTION ALGORITHMS 59
- 2.1 Transposition Ciphers 59
- 2.2 Simple Substitution Ciphers 62
  - 2.2.1 Single-Letter Frequency Analysis 66
- 2.3 Homophonic Substitution Ciphers 67
  - 2.3.1 Beale Ciphers 70

VMware Ex. 1016
Page 10 of 18

2.3.2 Higher-Order Homophonics 72
2.4 Polyalphabetic Substitution Ciphers 73
2.4.1 Vigenère and Beaufort Ciphers 74
2.4.2 Index of Coincidence 77
2.4.3 Kasiski Method 79
2.4.4 Running-Key Ciphers 83
2.4.5 Rotor and Hagelin Machines 84
2.4.6 Vernam Cipher and One-Time Pads 86
2.5 Polygram Substitution Ciphers 87
2.5.1 Playfair Cipher 87
2.5.2 Hill Cipher 88
2.6 Product Ciphers 90
2.6.1 Substitution-Permutation Ciphers 90
2.6.2 The Data Encryption Standard (DES) 92
2.6.3 Time-Memory Tradeoff 98
2.7 Exponentiation Ciphers 101
2.7.1 Pohlig-Hellman Scheme 103
2.7.2 Rivest-Shamir-Adleman (RSA) Scheme 104
2.7.3 Mental Poker 110
2.7.4 Oblivious Transfer 115
2.8 Knapsack Ciphers 117
2.8.1 Merkle-Hellman Knapsacks 118
2.8.2 Graham-Shamir Knapsacks 121
2.8.3 Shamir Signature-Only Knapsacks 122
2.8.4 A Breakable **NP**-Complete Knapsack 125
Exercises 126
References 129

3 CRYPTOGRAPHIC TECHNIQUES 135
3.1 Block and Stream Ciphers 135
3.2 Synchronous Stream Ciphers 138
3.2.1 Linear Feedback Shift Registers 139
3.2.2 Output-Block Feedback Mode 142
3.2.3 Counter Method 143
3.3 Self-Synchronous Stream Ciphers 144
3.3.1 Autokey Ciphers 145
3.3.2 Cipher Feedback 145
3.4 Block Ciphers 147
3.4.1 Block Chaining and Cipher Block Chaining 149
3.4.2 Block Ciphers with Subkeys 151
3.5 Endpoints of Encryption 154
3.5.1 End-to-End versus Link Encryption 154
3.5.2 Privacy Homomorphisms 157
3.6 One-Way Ciphers 161
3.6.1 Passwords and User Authentication 161

3.7 Key Management 164
3.7.1 Secret Keys 164
3.7.2 Public Keys 169
3.7.3 Generating Block Encryption Keys 171
3.7.4 Distribution of Session Keys 173
3.8 Threshold Schemes 179
3.8.1 Lagrange Interpolating Polynomial Scheme 180
3.8.2 Congruence Class Scheme 183
Exercises 185
References 187

4 ACCESS CONTROLS 191
4.1 Access-Matrix Model 192
4.1.1 The Protection State 192
4.1.2 State Transitions 194
4.1.3 Protection Policies 199
4.2 Access Control Mechanisms 200
4.2.1 Security and Precision 200
4.2.2 Reliability and Sharing 201
4.2.3 Design Principles 206
4.3 Access Hierarchies 207
4.3.1 Privileged Modes 207
4.3.2 Nested Program Units 208
4.4 Authorization Lists 209
4.4.1 Owned Objects 210
4.4.2 Revocation 213
4.5 Capabilities 216
4.5.1 Domain Switching with Protected Entry Points 218
4.5.2 Abstract Data Types 219
4.5.3 Capability-Based Addressing 224
4.5.4 Revocation 227
4.5.5 Locks and Keys 228
4.5.6 Query Modification 230
4.6 Verifiably Secure Systems 231
4.6.1 Security Kernels 232
4.6.2 Levels of Abstraction 235
4.6.3 Verification 236
4.7 Theory of Safe Systems 240
4.7.1 Mono-Operational Systems 241
4.7.2 General Systems 242
4.7.3 Theories for General Systems 245
4.7.4 Take-Grant Systems 248
Exercises 257
References 259

5 INFORMATION FLOW CONTROLS 265
5.1 Lattice Model of Information Flow 265
5.1.1 Information Flow Policy 265
5.1.2 Information State 266
5.1.3 State Transitions and Information Flow 267
5.1.4 Lattice Structure 273
5.1.5 Flow Properties of Lattices 276
5.2 Flow Control Mechanisms 279
5.2.1 Security and Precision 279
5.2.2 Channels of Flow 281
5.3 Execution-Based Mechanisms 282
5.3.1 Dynamically Enforcing Security for Implicit Flow 282
5.3.2 Flow-Secure Access Controls 285
5.3.3 Data Mark Machine 288
5.3.4 Single Accumulator Machine 290
5.4 Compiler-Based Mechanism 291
5.4.1 Flow Specifications 292
5.4.2 Security Requirements 293
5.4.3 Certification Semantics 297
5.4.4 General Data and Control Structures 298
5.4.5 Concurrency and Synchronization 302
5.4.6 Abnormal Terminations 305
5.5 Program Verification 307
5.5.1 Assignment 309
5.5.2 Compound 310
5.5.3 Alternation 311
5.5.4 Iteration 312
5.5.5 Procedure Call 313
5.5.6 Security 316
5.6 Flow Controls in Practice 318
5.6.1 System Verification 318
5.6.2 Extensions 320
5.6.3 A Guard Application 321
Exercises 324
References 327

6 INFERENCE CONTROLS 331
6.1 Statistical Database Model 332
6.1.1 Information State 332
6.1.2 Types of Statistics 334
6.1.3 Disclosure of Sensitive Statistics 336
6.1.4 Perfect Secrecy and Protection 339
6.1.5 Complexity of Disclosure 339
6.2 Inference Control Mechanisms 340
6.2.1 Security and Precision 340

6.2.2 Methods of Release 341
6.3 Methods of Attack 344
6.3.1 Small and Large Query Set Attacks 344
6.3.2 Tracker Attacks 346
6.3.3 Linear System Attacks 352
6.3.4 Median Attacks 356
6.3.5 Insertion and Deletion Attacks 358
6.4 Mechanisms that Restrict Statistics 358
6.4.1 Cell Suppression 360
6.4.2 Implied Queries 364
6.4.3 Partitioning 368
6.5 Mechanisms that Add Noise 371
6.5.1 Response Perturbation (Rounding) 372
6.5.2 Random-Sample Queries 374
6.5.3 Data Perturbation 380
6.5.4 Data Swapping 383
6.5.5 Randomized Response (Inquiry) 386
6.6 Summary 387
Exercises 388
References 390
INDEX 393

$$\{x_u \times y_v \mid v \neq u\} ,$$

which states that if x initially has the value u, then the new value of y cannot be v for any $v \neq u$. Thus the initial value of x can be deduced from the new value of y, and the security requirements of the statement are given by the condition

$$\underline{x} \leq \underline{y} . \quad \blacksquare$$

Example:
The specification statement

if $x = 0$ **then** $'y = 0$ **else** $'y = 1$

is described by the set of constraints

$$\{x_0 \times y_v \mid v \neq 0\} \cup \{x_u \times y_v \mid u \neq 0, v \neq 1\} ,$$

which shows that access to the new value of y can be used to deduce something about the initial value of x. The security requirements of this statement are given by the same condition as the previous statement. ■

The MITRE flow analyzer can handle arrays of elements belonging to different classes.

Example:
Consider the specification statement

if $x < 2$ **then** $'y = a[x]$.

This statement is described by the set of constraints

$$\{x_i\, a[i]_u \times y_v \mid v \neq u, i < 2\} .$$

The security requirements of the statement are thus given by

$$\underline{x} \leq \underline{y}$$
$$\underline{a[i]} \leq \underline{y} \text{ for } i < 2 . \quad \blacksquare$$

Rushby's [Rush81] approach to the design and verification of secure systems (see Section 4.6.3) seems to obviate the need to prove multilevel flow security for a security kernel. Since the kernel serves to isolate processes sharing the same machine, it suffices to prove such an isolation is achieved. Proofs of multilevel security are needed only for the explicit communication channels connecting processes.

5.6.2 Extensions

Practical systems often have information flow requirements that extend or conflict with the information flow models we have described thus far; for example,

1. **Integrity.** Information flow models describe the dissemination of information, but not its alteration. Thus an *Unclassified* process, for example, can write nonsense into a *Top Secret* file without violating the multilevel security policy. Although this problem is remedied by access controls, efforts to develop a multilevel integrity model have been pursued. Biba [Biba77] proposed a model where each object and process is assigned an integrity level, and a process cannot write into an object unless its integrity level is at least that of the object (in contrast, its security level must be no greater than that of the object).
2. **Sanitization and Downgrading.** Written documents routinely have their security classifications lowered ("downgraded"), and sensitive information is edited ("sanitized") for release at a lower level. Because these operations violate the multilevel security policy, they cannot be handled by the basic flow control mechanisms in a security kernel. The current practice is to place them in trusted processes that are permitted to violate the policies of the kernel. The trusted processes are verified (or at least audited) to show they meet the security requirements of the system as a whole. An example of such a trusted process is described in the next section.
3. **Aggregation.** An aggregate of data might require a higher classification than the individual items. This is because the aggregate might give an overall picture that cannot be deduced (or easily deduced) from an individual item. For example, the aggregate of all combat unit locations might reveal a plan of attack. This is the opposite of the inference problem discussed in the next chapter, where the problem is to provide a means of releasing statistics that give an overall picture, while protecting the individual values used to compute the statistics.

5.6.3 A Guard Application

The ACCAT Guard [Wood79] is an interesting blend of flow controls and cryptography. Developed by Logicon at the Advanced Command and Control Architectural Testbed (ACCAT), the Guard is a minicomputer interface between two computers (or networks of computers) of different classifications (called *Low* and *High*).

The Guard supports two types of communication: network mail and database transactions. The Guard allows information to flow without human intervention from *Low* to *High* computers, but approval by a Security Watch Officer is required for flows from *High* to *Low*. Figure 5.16 shows the role of the Security Watch Officer for both types of communication. For mail, information flows in one direction only, so human intervention is required only for mail transmitted from *High* to *Low* computers (Figure 5.16b). Database transactions always require human intervention, however, because information flows in one direction with the query and the reverse direction with the response. Queries from a *High* computer to a *Low* computer must pass through the Security Watch Officer,

VMware Ex. 1016
Page 17 of 18

ADDISON-WESLEY PUBLISHING COMPANY

ISBN 0-201-10150-5