

SECOND EDITION



COMPUTER SECURITY

[ART and SCIENCE]



MATT BISHOP



With contributions from ELISABETH SULLIVAN *and* MICHELLE RUPPEL

Computer Security

Second Edition

This page intentionally left blank

Computer Security

Art and Science

Second Edition

Matt Bishop
with contributions from
Elisabeth Sullivan and Michelle Ruppel

◆ Addison-Wesley

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

Many of the designations used by manufacturers and sellers 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 with initial capital letters or in all capitals.

The authors and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact intlcs@pearson.com.

Visit us on the Web: informit.com/aw

Library of Congress Control Number: 2018950017

Copyright © 2019 Pearson Education, Inc.

Cover illustration: Captiva55/Shutterstock

All rights reserved. This publication is protected by copyright, and permission must 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. For information regarding permissions, request forms and the appropriate contacts within the Pearson Education Global Rights & Permissions Department, please visit www.pearsoned.com/permissions/.

ISBN-13: 978-0-321-71233-2

ISBN-10: 0-321-71233-1

ScoutAutomatedPrintCode

To my dear Holly; our children Heidi, Steven, David, and Caroline; our grandchildren Skyler and Sage; and our friends Seaview, Tinker Belle, Stripe, Baby Windsor, Scout, Fur, Puff, Mouse, Shadow, Fuzzy, Dusty, and the rest of the menagerie.

This page intentionally left blank

Contents

Preface	xxix
Acknowledgments	xl
About the Author	xlix

PART I : INTRODUCTION **1**

Chapter 1 An Overview of Computer Security	3
1.1 The Basic Components	3
1.1.1 Confidentiality	4
1.1.2 Integrity	5
1.1.3 Availability	6
1.2 Threats	6
1.3 Policy and Mechanism	9
1.3.1 Goals of Security	10
1.4 Assumptions and Trust	11
1.5 Assurance	12
1.5.1 Specification	14
1.5.2 Design	14
1.5.3 Implementation	15
1.6 Operational Issues	16
1.6.1 Cost-Benefit Analysis	16
1.6.2 Risk Analysis	17
1.6.3 Laws and Customs	19
1.7 Human Issues	20
1.7.1 Organizational Problems	20
1.7.2 People Problems	21
1.8 Tying It All Together	22
1.9 Summary	24
1.10 Research Issues	24
1.11 Further Reading	25
1.12 Exercises	25

PART II : FOUNDATIONS**29**

Chapter 2	Access Control Matrix	31
2.1	Protection State	31
2.2	Access Control Matrix Model	32
2.2.1	Access Control by Boolean Expression Evaluation	35
2.2.2	Access Controlled by History	36
2.3	Protection State Transitions	37
2.3.1	Conditional Commands	40
2.4	Copying, Owning, and the Attenuation of Privilege	42
2.4.1	Copy Right	42
2.4.2	Own Right	42
2.4.3	Principle of Attenuation of Privilege	43
2.5	Summary	44
2.6	Research Issues	44
2.7	Further Reading	44
2.8	Exercises	45
Chapter 3	Foundational Results	49
3.1	The General Question	49
3.2	Basic Results	51
3.3	The Take-Grant Protection Model	56
3.3.1	Sharing of Rights	57
3.3.2	Interpretation of the Model	61
3.3.3	Theft in the Take-Grant Protection Model	62
3.3.4	Conspiracy	66
3.3.5	Summary	68
3.4	Closing the Gap: The Schematic Protection Model	68
3.4.1	Link Predicate	69
3.4.2	Filter Function	70
3.4.3	Putting It All Together	71
3.4.4	Demand and Create Operations	72
3.4.5	Safety Analysis	75
3.5	Expressive Power and the Models	81
3.5.1	Brief Comparison of HRU and SPM	82
3.5.2	Extending SPM	83
3.5.3	Simulation and Expressiveness	88
3.5.4	Typed Access Matrix Model	92
3.6	Comparing Security Properties of Models	94
3.6.1	Comparing Schemes and Security Properties	95
3.6.2	Augmented Typed Access Matrix Model	99

3.7	Summary	101
3.8	Research Issues	102
3.9	Further Reading	102
3.10	Exercises	103

PART III : POLICY **107**

Chapter 4 Security Policies **109**

4.1	The Nature of Security Policies	109
4.2	Types of Security Policies	113
4.3	The Role of Trust	115
4.4	Types of Access Control	117
4.5	Policy Languages	118
4.5.1	High-Level Policy Languages	119
4.5.2	Low-Level Policy Languages	125
4.6	Example: Academic Computer Security Policy	126
4.6.1	General University Electronic Communications Policy	127
4.6.2	Implementation at UC Davis	130
4.7	Security and Precision	131
4.8	Summary	136
4.9	Research Issues	136
4.10	Further Reading	137
4.11	Exercises	138

Chapter 5 Confidentiality Policies **141**

5.1	Goals of Confidentiality Policies	141
5.2	The Bell-LaPadula Model	142
5.2.1	Informal Description	142
5.2.2	Example: Trusted Solaris	146
5.2.3	Formal Model	151
5.2.4	Example Model Instantiation: Multics	158
5.3	Tranquility	161
5.3.1	Declassification Principles	163
5.4	The Controversy over the Bell-LaPadula Model	164
5.4.1	McLean's \dagger -Property and the Basic Security Theorem	164
5.4.2	McLean's System Z and More Questions	166
5.5	Summary	169
5.6	Research Issues	169
5.7	Further Reading	170
5.8	Exercises	171

Chapter 6 Integrity Policies	173
6.1 Goals	173
6.2 The Biba Model	175
6.2.1 Low-Water-Mark Policy	176
6.2.2 Ring Policy	177
6.2.3 Biba’s Model (Strict Integrity Policy)	177
6.3 Lipner’s Integrity Matrix Model	178
6.3.1 Lipner’s Use of the Bell-LaPadula Model	178
6.3.2 Lipner’s Full Model	181
6.3.3 Comparison with Biba	182
6.4 Clark-Wilson Integrity Model	183
6.4.1 The Model	184
6.4.2 Comparison with the Requirements	187
6.4.3 Comparison with Other Models	188
6.5 Trust Models	189
6.5.1 Policy-Based Trust Management	191
6.5.2 Reputation-Based Trust Management	194
6.6 Summary	196
6.7 Research Issues	196
6.8 Further Reading	197
6.9 Exercises	198
Chapter 7 Availability Policies	201
7.1 Goals of Availability Policies	201
7.2 Deadlock	202
7.3 Denial of Service Models	203
7.3.1 Constraint-Based Model	204
7.3.2 State-Based Modes	210
7.4 Example: Availability and Network Flooding	215
7.4.1 Analysis	216
7.4.2 Intermediate Systems	216
7.4.3 TCP State and Memory Allocations	218
7.4.4 Other Flooding Attacks	221
7.5 Summary	222
7.6 Research Issues	222
7.7 Further Reading	223
7.8 Exercises	224
Chapter 8 Hybrid Policies	227
8.1 Chinese Wall Model	227
8.1.1 Informal Description	228
8.1.2 Formal Model	230

8.1.3	Aggressive Chinese Wall Model	233
8.1.4	Bell-LaPadula and Chinese Wall Models	234
8.1.5	Clark-Wilson and Chinese Wall Models	236
8.2	Clinical Information Systems Security Policy	236
8.2.1	Bell-LaPadula and Clark-Wilson Models	239
8.3	Originator Controlled Access Control	239
8.3.1	Digital Rights Management	241
8.4	Role-Based Access Control	244
8.5	Break-the-Glass Policies	249
8.6	Summary	250
8.7	Research Issues	250
8.8	Further Reading	251
8.9	Exercises	252
Chapter 9 Noninterference and Policy Composition		255
9.1	The Problem	255
9.1.1	Composition of Bell-LaPadula Models	256
9.2	Deterministic Noninterference	259
9.2.1	Unwinding Theorem	263
9.2.2	Access Control Matrix Interpretation	266
9.2.3	Security Policies That Change over Time	268
9.2.4	Composition of Deterministic Noninterference-Secure Systems	270
9.3	Nondeducibility	271
9.3.1	Composition of Deducibly Secure Systems	273
9.4	Generalized Noninterference	274
9.4.1	Composition of Generalized Noninterference Systems	275
9.5	Restrictiveness	277
9.5.1	State Machine Model	277
9.5.2	Composition of Restrictive Systems	279
9.6	Side Channels and Deducibility	280
9.7	Summary	282
9.8	Research Issues	283
9.9	Further Reading	283
9.10	Exercises	285
PART IV : IMPLEMENTATION I: CRYPTOGRAPHY		287
Chapter 10 Basic Cryptography		289
10.1	Cryptography	289
10.1.1	Overview of Cryptanalysis	290

10.2	Symmetric Cryptosystems	291
10.2.1	Transposition Ciphers	291
10.2.2	Substitution Ciphers	292
10.2.3	Data Encryption Standard	299
10.2.4	Other Modern Symmetric Ciphers	302
10.2.5	Advanced Encryption Standard	303
10.3	Public Key Cryptography	306
10.3.1	El Gamal	307
10.3.2	RSA	309
10.3.3	Elliptic Curve Ciphers	312
10.4	Cryptographic Checksums	315
10.4.1	HMAC	317
10.5	Digital Signatures	318
10.5.1	Symmetric Key Signatures	319
10.5.2	Public Key Signatures	319
10.6	Summary	323
10.7	Research Issues	324
10.8	Further Reading	325
10.9	Exercises	326
 Chapter 11 Key Management		 331
11.1	Session and Interchange Keys	332
11.2	Key Exchange	332
11.2.1	Symmetric Cryptographic Key Exchange	333
11.2.2	Kerberos	337
11.2.3	Public Key Cryptographic Key Exchange and Authentication	338
11.3	Key Generation	341
11.4	Cryptographic Key Infrastructures	343
11.4.1	Merkle's Tree Authentication Scheme	344
11.4.2	Certificate Signature Chains	346
11.4.3	Public Key Infrastructures	350
11.5	Storing and Revoking Keys	353
11.5.1	Key Storage	353
11.5.2	Key Revocation	358
11.6	Summary	359
11.7	Research Issues	360
11.8	Further Reading	361
11.9	Exercises	362

Chapter 12	Cipher Techniques	367
12.1	Problems	367
12.1.1	Precomputing the Possible Messages	367
12.1.2	Misordered Blocks	368
12.1.3	Statistical Regularities	368
12.1.4	Type Flaw Attacks	369
12.1.5	Summary	370
12.2	Stream and Block Ciphers	370
12.2.1	Stream Ciphers	371
12.2.2	Block Ciphers	374
12.3	Authenticated Encryption	377
12.3.1	Counter with CBC-MAC Mode	377
12.3.2	Galois Counter Mode	379
12.4	Networks and Cryptography	381
12.5	Example Protocols	384
12.5.1	Secure Electronic Mail: PEM and OpenPGP	384
12.5.2	Instant Messaging	389
12.5.3	Security at the Transport Layer: TLS and SSL	393
12.5.4	Security at the Network Layer: IPsec	402
12.5.5	Conclusion	410
12.6	Summary	410
12.7	Research Issues	411
12.8	Further Reading	411
12.9	Exercises	413
Chapter 13	Authentication	415
13.1	Authentication Basics	415
13.2	Passwords	416
13.3	Password Selection	418
13.3.1	Random Selection of Passwords	418
13.3.2	Pronounceable and Other Computer-Generated Passwords	420
13.3.3	User Selection of Passwords	421
13.3.4	Graphical Passwords	425
13.4	Attacking Passwords	426
13.4.1	Off-Line Dictionary Attacks	428
13.4.2	On-Line Dictionary Attacks	430
13.4.3	Password Strength	432

13.5	Password Aging	434
13.5.1	One-Time Passwords	436
13.6	Challenge-Response	438
13.6.1	Pass Algorithms	438
13.6.2	Hardware-Supported Challenge-Response Procedures	439
13.6.3	Challenge-Response and Dictionary Attacks	439
13.7	Biometrics	441
13.7.1	Fingerprints	442
13.7.2	Voices	443
13.7.3	Eyes	443
13.7.4	Faces	444
13.7.5	Keystrokes	444
13.7.6	Combinations	445
13.8	Location	445
13.9	Multifactor Authentication	446
13.10	Summary	448
13.11	Research Issues	449
13.12	Further Reading	450
13.13	Exercises	451

PART V : IMPLEMENTATION II: SYSTEMS 453

Chapter 14 Design Principles 455

14.1	Underlying Ideas	455
14.2	Principles of Secure Design	457
14.2.1	Principle of Least Privilege	457
14.2.2	Principle of Fail-Safe Defaults	458
14.2.3	Principle of Economy of Mechanism	459
14.2.4	Principle of Complete Mediation	460
14.2.5	Principle of Open Design	461
14.2.6	Principle of Separation of Privilege	463
14.2.7	Principle of Least Common Mechanism	463
14.2.8	Principle of Least Astonishment	464
14.3	Summary	466
14.4	Research Issues	466
14.5	Further Reading	467
14.6	Exercises	468

Chapter 15 Representing Identity 471

15.1	What Is Identity?	471
15.2	Files and Objects	472
15.3	Users	473

15.4	Groups and Roles	475
15.5	Naming and Certificates	476
15.5.1	Conflicts	479
15.5.2	The Meaning of the Identity	481
15.5.3	Trust	482
15.6	Identity on the Web	484
15.6.1	Host Identity	484
15.6.2	State and Cookies	488
15.7	Anonymity on the Web	490
15.7.1	Email Anonymizers	491
15.7.2	Onion Routing	495
15.8	Summary	501
15.9	Research Issues	502
15.10	Further Reading	503
15.11	Exercises	504
Chapter 16 Access Control Mechanisms		507
16.1	Access Control Lists	507
16.1.1	Abbreviations of Access Control Lists	508
16.1.2	Creation and Maintenance of Access Control Lists	511
16.1.3	Revocation of Rights	514
16.1.4	Example: NTFS and Access Control Lists	515
16.2	Capabilities	518
16.2.1	Implementation of Capabilities	519
16.2.2	Copying and Amplifying Capabilities	520
16.2.3	Revocation of Rights	522
16.2.4	Limits of Capabilities	522
16.2.5	Comparison with Access Control Lists	523
16.2.6	Privileges	524
16.3	Locks and Keys	526
16.3.1	Type Checking	528
16.3.2	Sharing Secrets	529
16.4	Ring-Based Access Control	531
16.5	Propagated Access Control Lists	533
16.6	Summary	535
16.7	Research Issues	535
16.8	Further Reading	536
16.9	Exercises	536
Chapter 17 Information Flow		539
17.1	Basics and Background	539
17.1.1	Entropy-Based Analysis	540
17.1.2	Information Flow Models and Mechanisms	541

- 17.2 Nonlattice Information Flow Policies 542
 - 17.2.1 Confinement Flow Model 543
 - 17.2.2 Transitive Nonlattice Information Flow Policies 544
 - 17.2.3 Nontransitive Information Flow Policies 545
- 17.3 Static Mechanisms 548
 - 17.3.1 Declarations 549
 - 17.3.2 Program Statements 550
 - 17.3.3 Exceptions and Infinite Loops 557
 - 17.3.4 Concurrency 558
 - 17.3.5 Soundness 561
- 17.4 Dynamic Mechanisms 562
 - 17.4.1 Fenton’s Data Mark Machine 562
 - 17.4.2 Variable Classes 565
- 17.5 Integrity Mechanisms 566
- 17.6 Example Information Flow Controls 567
 - 17.6.1 Privacy and Android Cell Phones 568
 - 17.6.2 Firewalls 570
- 17.7 Summary 574
- 17.8 Research Issues 574
- 17.9 Further Reading 575
- 17.10 Exercises 576

Chapter 18 Confinement Problem 579

- 18.1 The Confinement Problem 579
- 18.2 Isolation 582
 - 18.2.1 Controlled Environment 582
 - 18.2.2 Program Modification 590
- 18.3 Covert Channels 594
 - 18.3.1 Detection of Covert Channels 596
 - 18.3.2 Analysis of Covert Channels 610
 - 18.3.3 Mitigation of Covert Channels 616
- 18.4 Summary 619
- 18.5 Research Issues 620
- 18.6 Further Reading 620
- 18.7 Exercises 622

PART VI : ASSURANCE 625

Contributed by Elisabeth Sullivan and Michelle Ruppel

Chapter 19 Introduction to Assurance 627

- 19.1 Assurance and Trust 627
 - 19.1.1 The Need for Assurance 629

19.1.2	The Role of Requirements in Assurance	631
19.1.3	Assurance throughout the Life Cycle	632
19.2	Building Secure and Trusted Systems	634
19.2.1	Life Cycle	634
19.2.2	The Waterfall Life Cycle Model	639
19.2.3	Agile Software Development	641
19.2.4	Other Models of Software Development	644
19.3	Summary	645
19.4	Research Issues	645
19.5	Further Reading	646
19.6	Exercises	647
Chapter 20 Building Systems with Assurance		649
20.1	Assurance in Requirements Definition and Analysis	649
20.1.1	Threats and Security Objectives	650
20.1.2	Architectural Considerations	651
20.1.3	Policy Definition and Requirements Specification	657
20.1.4	Justifying Requirements	660
20.2	Assurance during System and Software Design	662
20.2.1	Design Techniques That Support Assurance	662
20.2.2	Design Document Contents	665
20.2.3	Building Documentation and Specification	675
20.2.4	Justifying That Design Meets Requirements	677
20.3	Assurance in Implementation and Integration	685
20.3.1	Implementation Considerations That Support Assurance	685
20.3.2	Assurance through Implementation Management	686
20.3.3	Justifying That the Implementation Meets the Design	687
20.4	Assurance during Operation and Maintenance	695
20.5	Summary	696
20.6	Research Issues	696
20.7	Further Reading	697
20.8	Exercises	698
Chapter 21 Formal Methods		699
21.1	Formal Verification Techniques	699
21.2	Formal Specification	702
21.3	Early Formal Verification Techniques	705
21.3.1	The Hierarchical Development Methodology	705
21.3.2	Enhanced HDM	710
21.3.3	The Gypsy Verification Environment	711

21.4	Current Verification Systems	713
21.4.1	The Prototype Verification System	713
21.4.2	The Symbolic Model Verifier	716
21.4.3	The Naval Research Laboratory Protocol Analyzer	720
21.5	Functional Programming Languages	721
21.6	Formally Verified Products	722
21.7	Summary	723
21.8	Research Issues	724
21.9	Further Reading	725
21.10	Exercises	725
Chapter 22 Evaluating Systems		727
22.1	Goals of Formal Evaluation	727
22.1.1	Deciding to Evaluate	728
22.1.2	Historical Perspective of Evaluation Methodologies	729
22.2	TCSEC: 1983–1999	730
22.2.1	TCSEC Requirements	731
22.2.2	The TCSEC Evaluation Classes	733
22.2.3	The TCSEC Evaluation Process	734
22.2.4	Impacts	735
22.3	International Efforts and the ITSEC: 1991–2001	737
22.3.1	ITSEC Assurance Requirements	739
22.3.2	The ITSEC Evaluation Levels	740
22.3.3	The ITSEC Evaluation Process	741
22.3.4	Impacts	741
22.4	Commercial International Security Requirements: 1991	742
22.4.1	CISR Requirements	743
22.4.2	Impacts	743
22.5	Other Commercial Efforts: Early 1990s	744
22.6	The Federal Criteria: 1992	744
22.6.1	FC Requirements	745
22.6.2	Impacts	745
22.7	FIPS 140: 1994–Present	746
22.7.1	FIPS 140 Requirements	746
22.7.2	FIPS 140-2 Security Levels	747
22.7.3	Additional FIPS 140-2 Documentation	748
22.7.4	Impact	748
22.7.5	Future	749
22.8	The Common Criteria: 1998–Present	749
22.8.1	Overview of the Methodology	751
22.8.2	CC Requirements	756
22.8.3	CC Security Functional Requirements	756

22.8.4	Assurance Requirements	759
22.8.5	Evaluation Assurance Levels	759
22.8.6	Evaluation Process	761
22.8.7	Other International Organizations	762
22.8.8	Impacts	763
22.8.9	Future of the Common Criteria	764
22.9	SSE-CMM: 1997–Present	765
22.9.1	The SSE-CMM Model	765
22.9.2	Using the SSE-CMM	767
22.10	Summary	768
22.11	Research Issues	769
22.12	Further Reading	769
22.13	Exercises	770

PART VII : SPECIAL TOPICS 773

Chapter 23	Malware	775
23.1	Introduction	775
23.2	Trojan Horses	776
23.2.1	Rootkits	777
23.2.2	Propagating Trojan Horses	779
23.3	Computer Viruses	780
23.3.1	Infection Vectors	782
23.3.2	Concealment	785
23.3.3	Summary	790
23.4	Computer Worms	790
23.5	Bots and Botnets	793
23.6	Other Malware	796
23.6.1	Rabbits and Bacteria	796
23.6.2	Logic Bombs	797
23.6.3	Adware	797
23.6.4	Spyware	799
23.6.5	Ransomware	800
23.6.6	Phishing	802
23.7	Combinations	803
23.8	Theory of Computer Viruses	803
23.9	Defenses	808
23.9.1	Scanning Defenses	808
23.9.2	Data and Instructions	811
23.9.3	Containment	812
23.9.4	Specifications as Restrictions	817

23.9.5	Limiting Sharing	817
23.9.6	Statistical Analysis	819
23.9.7	The Notion of Trust	819
23.10	Summary	820
23.11	Research Issues	820
23.12	Further Reading	821
23.13	Exercises	822
Chapter 24 Vulnerability Analysis		825
24.1	Introduction	825
24.2	Penetration Studies	827
24.2.1	Goals	827
24.2.2	Layering of Tests	828
24.2.3	Methodology at Each Layer	829
24.2.4	Flaw Hypothesis Methodology	830
24.2.5	Versions	833
24.2.6	Example: Penetration of the Michigan Terminal System	837
24.2.7	Example: Compromise of a Burroughs System	839
24.2.8	Example: Penetration of a Corporate Computer System . . .	840
24.2.9	Example: Penetrating a UNIX System	841
24.2.10	Example: Penetrating a Windows System	843
24.2.11	Debate	844
24.2.12	Conclusion	845
24.3	Vulnerability Classification	845
24.3.1	Two Security Flaws	846
24.4	Frameworks	849
24.4.1	The RISOS Study	849
24.4.2	Protection Analysis Model	851
24.4.3	The NRL Taxonomy	857
24.4.4	Aslam's Model	859
24.4.5	Comparison and Analysis	860
24.5	Standards	864
24.5.1	Common Vulnerabilities and Exposures (CVE)	864
24.5.2	Common Weaknesses and Exposures (CWE)	866
24.6	Gupta and Gligor's Theory of Penetration Analysis	868
24.6.1	The Flow-Based Model of Penetration Analysis	869
24.6.2	The Automated Penetration Analysis Tool	872
24.6.3	Discussion	873
24.7	Summary	873
24.8	Research Issues	874
24.9	Further Reading	875
24.10	Exercises	876

Chapter 25 Auditing	879
25.1 Definition	879
25.2 Anatomy of an Auditing System	880
25.2.1 Logger	881
25.2.2 Analyzer	883
25.2.3 Notifier	883
25.3 Designing an Auditing System	884
25.3.1 Implementation Considerations	886
25.3.2 Syntactic Issues	887
25.3.3 Log Sanitization	888
25.3.4 Application and System Logging	891
25.4 A Posteriori Design	893
25.4.1 Auditing to Detect Violations of a Known Policy	893
25.4.2 Auditing to Detect Known Violations of a Policy	895
25.5 Auditing Mechanisms	897
25.5.1 Secure Systems	897
25.5.2 Nonsecure Systems	899
25.6 Examples: Auditing File Systems	900
25.6.1 Audit Analysis of the NFS Version 2 Protocol	900
25.6.2 The Logging and Auditing File System (LAFS)	905
25.6.3 Comparison	907
25.6.4 Audit Browsing	908
25.7 Summary	910
25.8 Research Issues	911
25.9 Further Reading	912
25.10 Exercises	913
Chapter 26 Intrusion Detection	917
26.1 Principles	917
26.2 Basic Intrusion Detection	918
26.3 Models	920
26.3.1 Anomaly Modeling	920
26.3.2 Misuse Modeling	932
26.3.3 Specification Modeling	938
26.3.4 Summary	941
26.4 Architecture	942
26.4.1 Agent	943
26.4.2 Director	945
26.4.3 Notifier	946
26.5 Organization of Intrusion Detection Systems	948
26.5.1 Monitoring Network Traffic for Intrusions: NSM	948
26.5.2 Combining Host and Network Monitoring: DIDS	949
26.5.3 Autonomous Agents: AAFID	952

26.6 Summary 954
 26.7 Research Issues 954
 26.8 Further Reading 955
 26.9 Exercises 956

Chapter 27 Attacks and Responses 959

27.1 Attacks 959
 27.2 Representing Attacks 960
 27.2.1 Attack Trees 961
 27.2.2 The Requires/Provides Model 965
 27.2.3 Attack Graphs 969
 27.3 Intrusion Response 971
 27.3.1 Incident Prevention 971
 27.3.2 Intrusion Handling 975
 27.4 Digital Forensics 987
 27.4.1 Principles 987
 27.4.2 Practice 990
 27.4.3 Anti-Forensics 994
 27.5 Summary 996
 27.6 Research Issues 997
 27.7 Further Reading 998
 27.8 Exercises 999

PART VIII : PRACTICUM 1003

Chapter 28 Network Security 1005

28.1 Introduction 1005
 28.2 Policy Development 1006
 28.2.1 Data Classes 1007
 28.2.2 User Classes 1008
 28.2.3 Availability 1010
 28.2.4 Consistency Check 1010
 28.3 Network Organization 1011
 28.3.1 Analysis of the Network Infrastructure 1013
 28.3.2 In the DMZ 1017
 28.3.3 In the Internal Network 1021
 28.3.4 General Comment on Assurance 1025
 28.4 Availability 1026

28.5	Anticipating Attacks	1027
28.6	Summary	1028
28.7	Research Issues	1028
28.8	Further Reading	1029
28.9	Exercises	1030
Chapter 29 System Security		1035
29.1	Introduction	1035
29.2	Policy	1036
29.2.1	The WWW Server System in the DMZ	1036
29.2.2	The Development System	1037
29.2.3	Comparison	1041
29.2.4	Conclusion	1041
29.3	Networks	1042
29.3.1	The WWW Server System in the DMZ	1042
29.3.2	The Development System	1045
29.3.3	Comparison	1047
29.4	Users	1048
29.4.1	The WWW Server System in the DMZ	1048
29.4.2	The Development System	1050
29.4.3	Comparison	1052
29.5	Authentication	1053
29.5.1	The WWW Server System in the DMZ	1053
29.5.2	Development Network System	1054
29.5.3	Comparison	1055
29.6	Processes	1055
29.6.1	The WWW Server System in the DMZ	1055
29.6.2	The Development System	1059
29.6.3	Comparison	1060
29.7	Files	1061
29.7.1	The WWW Server System in the DMZ	1061
29.7.2	The Development System	1063
29.7.3	Comparison	1065
29.8	Retrospective	1066
29.8.1	The WWW Server System in the DMZ	1066
29.8.2	The Development System	1067
29.9	Summary	1068
29.10	Research Issues	1068
29.11	Further Reading	1069
29.12	Exercises	1070

Chapter 30	User Security	1073
30.1	Policy	1073
30.2	Access	1074
30.2.1	Passwords	1074
30.2.2	The Login Procedure	1076
30.2.3	Leaving the System	1079
30.3	Files and Devices	1080
30.3.1	Files	1080
30.3.2	Devices	1084
30.4	Processes	1087
30.4.1	Copying and Moving Files	1087
30.4.2	Accidentally Overwriting Files	1088
30.4.3	Encryption, Cryptographic Keys, and Passwords	1089
30.4.4	Startup Settings	1090
30.4.5	Limiting Privileges	1091
30.4.6	Malicious Logic	1091
30.5	Electronic Communications	1092
30.5.1	Automated Electronic Mail Processing	1092
30.5.2	Failure to Check Certificates	1093
30.5.3	Sending Unexpected Content	1094
30.6	Summary	1094
30.7	Research Issues	1095
30.8	Further Reading	1095
30.9	Exercises	1096
Chapter 31	Program Security	1099
31.1	Problem	1099
31.2	Requirements and Policy	1100
31.2.1	Requirements	1100
31.2.2	Threats	1102
31.3	Design	1104
31.3.1	Framework	1104
31.3.2	Access to Roles and Commands	1106
31.4	Refinement and Implementation	1111
31.4.1	First-Level Refinement	1111
31.4.2	Second-Level Refinement	1112
31.4.3	Functions	1114
31.4.4	Summary	1117
31.5	Common Security-Related Programming Problems	1117
31.5.1	Improper Choice of Initial Protection Domain	1118
31.5.2	Improper Isolation of Implementation Detail	1123

31.5.3	Improper Change	1125
31.5.4	Improper Naming	1129
31.5.5	Improper Deallocation or Deletion	1131
31.5.6	Improper Validation	1132
31.5.7	Improper Indivisibility	1138
31.5.8	Improper Choice of Operand or Operation	1139
31.5.9	Summary	1141
31.6	Testing, Maintenance, and Operation	1141
31.6.1	Testing	1142
31.6.2	Testing Composed Modules	1145
31.6.3	Testing the Program	1145
31.7	Distribution	1146
31.8	Summary	1147
31.9	Research Issues	1147
31.10	Further Reading	1148
31.11	Exercises	1148

PART IX : APPENDICES 1151

Appendix A Lattices 1153

A.1	Basics	1153
A.2	Lattices	1154
A.3	Exercises	1155

Appendix B The Extended Euclidean Algorithm 1157

B.1	The Euclidean Algorithm	1157
B.2	The Extended Euclidean Algorithm	1158
B.3	Solving $ax \bmod n = 1$	1160
B.4	Solving $ax \bmod n = b$	1161
B.5	Exercises	1161

Appendix C Entropy and Uncertainty 1163

C.1	Conditional and Joint Probability	1163
C.2	Entropy and Uncertainty	1165
C.3	Joint and Conditional Entropy	1166
C.3.1	Joint Entropy	1166
C.3.2	Conditional Entropy	1167
C.3.3	Perfect Secrecy	1168
C.4	Exercises	1169

Appendix D	Virtual Machines	1171
D.1	Virtual Machine Structure	1171
D.2	Virtual Machine Monitor	1171
D.2.1	Privilege and Virtual Machines	1172
D.2.2	Physical Resources and Virtual Machines	1175
D.2.3	Paging and Virtual Machines	1175
D.3	Exercises	1176
Appendix E	Symbolic Logic	1179
E.1	Propositional Logic	1179
E.1.1	Natural Deduction in Propositional Logic	1180
E.1.2	Rules	1180
E.1.3	Derived Rules	1181
E.1.4	Well-Formed Formulas	1182
E.1.5	Truth Tables	1182
E.1.6	Mathematical Induction	1183
E.2	Predicate Logic	1184
E.2.1	Natural Deduction in Predicate Logic	1185
E.3	Temporal Logic Systems	1186
E.3.1	Syntax of CTL	1186
E.3.2	Semantics of CTL	1186
E.4	Exercises	1188
Appendix F	The Encryption Standards	1191
F.1	Data Encryption Standard	1191
F.1.1	Main DES Algorithm	1191
F.1.2	Round Key Generation	1195
F.2	Advanced Encryption Standard	1196
F.2.1	Background	1196
F.2.2	AES Encryption	1197
F.2.3	AES Decryption	1200
F.2.4	Round Key Generation	1201
F.2.5	Equivalent Inverse Cipher Implementation	1203
F.3	Exercises	1205
Appendix G	Example Academic Security Policy	1207
G.1	Acceptable Use Policy	1207
G.1.1	Introduction	1208
G.1.2	Rights and Responsibilities	1208
G.1.3	Privacy	1208

G.1.4	Enforcement of Laws and University Policies	1209
G.1.5	Unacceptable Conduct	1209
G.1.6	Further Information	1212
G.2	University of California Electronic Communications Policy	1212
G.2.1	Introduction	1212
G.2.2	General Provisions	1213
G.2.3	Allowable Use	1216
G.2.4	Privacy and Confidentiality	1220
G.2.5	Security	1225
G.2.6	Retention and Disposition	1227
G.2.7	Appendix A: Definitions	1227
G.2.8	Appendix B: References	1230
G.2.9	Appendix C: Policies Relating to Access Without Consent	1232
G.3	User Advisories	1234
G.3.1	Introduction	1234
G.3.2	User Responsibilities	1234
G.3.3	Privacy Expectations	1235
G.3.4	Privacy Protections	1236
G.3.5	Privacy Limits	1237
G.3.6	Security Considerations	1239
G.4	Electronic Communications—Allowable Use	1241
G.4.1	Purpose	1241
G.4.2	Definitions	1242
G.4.3	Policy	1242
G.4.4	Allowable Users	1242
G.4.5	Allowable Uses	1243
G.4.6	Restrictions on Use	1245
G.4.7	References and Related Policies	1246
	Appendix H Programming Rules	1247
H.1	Implementation Rules	1247
H.2	Management Rules	1249
	References	1251
	Index	1341

This page intentionally left blank

Preface

HORTENSIO: Madam, before you touch the instrument
To learn the order of my fingering,
I must begin with rudiments of art
To teach you gamouth in a briefer sort,
More pleasant, pithy and effectual,
Than hath been taught by any of my trade;
And there it is in writing, fairly drawn.
— *The Taming of the Shrew*, III, i, 62–68.

Preface to the Second Edition

Since the first edition of this book was published, the number of computer and information security incidents has increased dramatically, as has their seriousness. In 2010, a computer worm infected the software controlling a particular type of centrifuge used in uranium-enrichment sites [1116, 1137]. In 2013, a security breach at Target, a large chain of stores in the United States, compromised 40 million credit cards [1497, 1745, 2237]. Also in 2013, Yahoo reported that an attack compromised more than 1 billion accounts [779]. In 2017, attackers spread ransomware that crippled computers throughout the world, including computers used in hospitals and telecommunications companies [1881]. Equifax estimated that attackers also compromised the personal data of over 100,000,000 people [176].

These attacks exploit vulnerabilities that have their roots in vulnerabilities of the 1980s, 1970s, and earlier. They seem more complex because systems have become more complex, and thus the vulnerabilities are more obscure and require more complex attacks to exploit. But the principles underlying the attacks, the vulnerabilities, and the failures of the systems have not changed—only the arena in which they are applied has.

Consistent with this philosophy, the second edition continues to focus on the principles underlying the field of computer and information security. Many

newer examples show how these principles are applied, or not applied, today; but the principles themselves are as important today as they were in 2002, and earlier. Some have been updated to reflect a deeper understanding of people and systems. Others have been applied in new and interesting ways. But they still ring true.

That said, the landscape of security has evolved greatly in the years since this book was first published. The explosive growth of the World Wide Web, and the consequent explosion in its use, has made security a problem at the forefront of our society. No longer can vulnerabilities, both human and technological, be relegated to the background of our daily lives. It is one of the elements at the forefront, playing a role in everyone's life as one browses the web, uses a camera to take and send pictures, and turns on an oven remotely. We grant access to our personal lives through social media such as Facebook, Twitter, and Instagram, and to our homes through the Internet of Things and our connections to the Internet. To ignore security issues, or consider them simply ancillary details that "someone will fix somehow" or threats unlikely to be realized personally is dangerous at best, and potentially disastrous at worst.

Ultimately, little has changed. The computing ecosystem of our day is badly flawed. Among the manifestations of these technological flaws are that security problems continue to exist, and continue to grow in magnitude of effect. An interesting question to ponder is what might move the paradigm of security away from the cycle of "patch and catch" and "let the buyer beware" to a stable and safer ecosystem.

But we must continue to improve our understanding of, and implementation of, security. Security nihilism—simply giving up and asserting that we cannot make things secure, so why try—means we accept these invasions of our privacy, our society, and our world. Like everything else, security is imperfect, and always will be—meaning we can improve the state of the art. This book is directed towards that goal.

Updated Roadmap

The dependencies of the chapters are the same as in the first edition (see p. xl), with two new chapters added.

Chapter 7, which includes a discussion of denial of service attack models, contains material useful for Chapters 23, 24, 27, and 28. Similarly, Chapter 27 draws on material from the chapters in Part III as well as Chapters 23, 25, 26, and all of Part VIII.

In addition to the suggestions in the preface to the first edition on p. xli about topics for undergraduate classes, the material in Chapter 27 will introduce undergraduates to how attacks occur, how they can be analyzed, and what their effects are. Coupled with current examples drawn from the news, this chapter should prove fascinating to undergraduates.

As for graduate classes, the new material in Chapter 7 will provide students with some background on resilience, a topic increasing in importance. Otherwise, the recommendations are the same as for the first edition (see p. xlii).

Changes to the First Edition

The second edition has extensively revised many examples to apply the concepts to technologies, methodologies, and ideas that have emerged since the first edition was published. Here, the focus is on new material in the chapters; changes to examples are mentioned only when necessary to describe that material. In addition to what is mentioned here, much of the text has been updated.

Chapter 1, “An Overview of Computer Security”: This chapter is largely unchanged.

Chapter 2, “Access Control Matrix”: Section 2.2.2, “Access Controlled by History” has been changed to use the problem of preventing downloaded programs from accessing the system in unauthorized ways, instead of updating a database. Section 2.4.3, “Principle of Attenuation of Privilege,” has been expanded slightly, and exercises added to point out differing forms of the principle.

Chapter 3, “Foundational Results”: Definition 3–1 has been updated to make clear that “leaking” refers to a right being added to an element of the access control matrix that did not contain it initially, and an exercise has been added to demonstrate the difference between this definition and the one in the first edition. Section 3.6 discusses comparing security properties of models.

Chapter 4, “Security Policies”: Section 4.5.1, “High-Level Policy Languages,” now uses Ponder rather than a Java policy constraint language. Section 4.6, “Example: Academic Computer Security Policy,” has been updated to reflect changes in the university policy.

Chapter 5, “Confidentiality Policies”: Section 5.3.1 discusses principles for declassifying information.

Chapter 6, “Integrity Policies”: Section 6.5 presents trust models.

Chapter 7, “Availability Policies”: This chapter is new.

Chapter 8, “Hybrid Policies”: Section 8.1.3 modifies one of the assumptions of the Chinese Wall model that is unrealistic. Section 8.3.1 expands the discussion of ORCON to include DRM. Section 8.4 adds a discussion of several types of RBAC models.

Chapter 9, “Noninterference and Policy Composition”: This chapter adds Section 9.6, which presents side channels in the context of deducibility.

Chapter 10, “Basic Cryptography”: This chapter has been extensively revised. The discussion of the DES (Section 10.2.3) has been tightened and the algorithm

moved to Appendix F. Discussions of the AES (Section 10.2.5) and elliptic curve cryptography (Section 10.3.3) have been added, and the section on digital signatures moved from Chapter 11 to Section 10.5. Also, the number of digits in the integers used in examples for public key cryptography has been increased from 2 to at least 4, and in many cases more.

Chapter 11, “Key Management”: Section 11.4.3 discusses public key infrastructures. Section 11.5.1.4, “Other Approaches,” now includes a brief discussion of identity-based encryption.

Chapter 12, “Cipher Techniques”: Section 12.1, “Problems,” now includes a discussion of type flaw attacks. Section 12.3 discusses authenticated encryption with associated data, and presents the CCM and GCM modes of block ciphers. A new section, Section 12.5.2, discusses the Signal Protocol. Section 12.5.3, “Security at the Transport Layer: TLS and SSL,” has been expanded and focuses on TLS rather than SSL. It also discusses cryptographic weaknesses in SSL, such as the POODLE attack, that have led to the use of SSL being strongly discouraged.

Chapter 13, “Authentication”: A discussion of graphical passwords has been added as Section 13.3.4. Section 13.4.3 looks at quantifying password strength in terms of entropy. The discussion of biometrics in Section 13.7 has been expanded to reflect their increasing use.

Chapter 14, “Design Principles”: The principle of least authority follows the principle of least privilege in Section 14.2.1, and the principle of least astonishment now supersedes the principle of psychological acceptability in Section 14.2.8.

Chapter 15, “Representing Identity”: Section 15.5, “Naming and Certificates,” now includes a discussion of registration authorities (RAs). Section 15.6.1.3 adds a discussion of the DNS security extensions (DNSSEC). Section 15.7.2 discusses onion routing and Tor in the context of anonymity.

Chapter 16, “Access Control Mechanisms”: Section 16.2.6 discusses sets of privileges in Linux and other UNIX-like systems.

Chapter 17, “Information Flow”: In contrast to the confidentiality-based context of information flow in the main part of this chapter, Section 17.5 presents information flow in an integrity context. In Section 17.6, the SPI and SNSMG examples of the first edition have been replaced by Android cell phones (Section 17.6.1) and firewalls (Section 17.6.2).

Chapter 18, “Confinement Problem”: Section 18.2 has been expanded to include library operating systems (Section 18.2.1.2) and program modification techniques (Section 18.2.2).

Chapter 19, “Introduction to Assurance”: Section 19.2.3, which covers agile software development, has been added.

Chapter 20, “Building Systems with Assurance”: The example decomposition of Windows 2000 into components has been updated to use Windows 10.

Chapter 21, “Formal Methods”: A new section, Section 21.5, discusses functional programming languages, and another new section, 21.6, discusses formally verified products.

Chapter 22, “Evaluating Systems”: Sections 22.7, on FIPS 140, and 22.8, on the Common Criteria, have been extensively updated.

Chapter 23, “Malware”: Section 23.5 presents botnets, and Sections 23.6.3, 23.6.4, 23.6.5, and 23.6.6 discuss adware and spyware, ransomware, and phishing. While not malware, phishing is a common vector for getting malware onto a system and so it is discussed here.

Chapter 24, “Vulnerability Analysis”: Section 24.2.5 reviews several penetration testing frameworks used commercially and based on the Flaw Hypothesis Methodology. Section 24.5 presents the widely used CVE and CWE standards.

Chapter 25, “Auditing”: Section 25.3.3, which discusses sanitization, has been expanded.

Chapter 26, “Intrusion Detection”: Section 26.3.1 has been expanded to include several widely used machine learning techniques for anomaly detection. Incident response groups are discussed in Section 27.3.

Chapter 27, “Attacks and Responses”: This chapter is new.

Chapter 28, “Network Security”: The discussion of what firewalls are has been moved to Section 17.6.2, but the discussion of how the Drib configures and uses them remains in this chapter. The Drib added wireless networks, which are discussed in Section 28.3.3.1. Its analysis of using the cloud is in Section 28.3.3.2. The rest of the chapter has been updated to refer to the new material in previous chapters.

Chapter 29, “System Security”: This chapter has been updated to refer to the new material in previous chapters.

Chapter 30, “User Security”: Section 30.2.2 describes the two-factor authentication procedure used by the Drib. The rest of the chapter has been updated to refer to the new material in previous chapters.

Chapter 31, “Program Security”: This chapter has been updated to refer to the new material in previous chapters.

Two new appendices have been added. Appendix F presents the DES and AES algorithms, and Appendix H collects the rules in Chapter 31 for easy reference. In addition, Appendix D examines some hardware enhancements to aid virtualization, and Appendix G contains the full academic security policy discussed in Section 4.6.

Preface to the First Edition¹

On September 11, 2001, terrorists seized control of four airplanes. Three were flown into buildings, and a fourth crashed, with catastrophic loss of life. In the aftermath, the security and reliability of many aspects of society drew renewed scrutiny. One of these aspects was the widespread use of computers and their interconnecting networks. The issue is not new. In 1988, approximately 5,000 computers throughout the Internet were rendered unusable within 4 hours by a program called a worm [842].² While the spread, and the effects, of this program alarmed computer scientists, most people were not worried because the worm did not affect their lives or their ability to do their jobs. In 1993, more users of computer systems were alerted to such dangers when a set of programs called sniffers were placed on many computers run by network service providers and recorded login names and passwords [670].

After an attack on Tsutomu Shimomura's computer system, and the fascinating way Shimomura followed the attacker's trail, which led to his arrest [1736], the public's interest and apprehension were finally aroused. Computers were now vulnerable. Their once reassuring protections were now viewed as flimsy.

Several films explored these concerns. Movies such as *War Games* and *Hackers* provided images of people who can, at will, wander throughout computers and networks, maliciously or frivolously corrupting or destroying information it may have taken millions of dollars to amass. (Reality intruded on *Hackers* when the World Wide Web page set up by MGM/United Artists was quickly altered to present an irreverent commentary on the movie and to suggest that viewers see *The Net* instead. Paramount Pictures denied doing this [869].) Another film, *Sneakers*, presented a picture of those who test the security of computer (and other) systems for their owners and for the government.

Goals

This book has three goals. The first is to show the importance of theory to practice and of practice to theory. All too often, practitioners regard theory as irrelevant and theoreticians think of practice as trivial. In reality, theory and practice are symbiotic. For example, the theory of covert channels, in which the goal is to limit the ability of processes to communicate through shared resources, provides a mechanism for evaluating the effectiveness of mechanisms that confine processes, such as sandboxes and firewalls. Similarly, business practices in the commercial world led to the development of several security policy models such as the Clark-Wilson model and the Chinese Wall model. These models in turn help the designers of security policies better understand and evaluate the mechanisms and procedures needed to secure their sites.

¹Chapter numbers have been updated to correspond to the chapters in the second edition.

²Section 23.4 discusses computer worms.

The second goal is to emphasize that computer security and cryptography are different. Although cryptography is an essential component of computer security, it is by no means the only component. Cryptography provides a mechanism for performing specific functions, such as preventing unauthorized people from reading and altering messages on a network. However, unless developers understand the context in which they are using cryptography, and unless the assumptions underlying the protocol and the cryptographic mechanisms apply to the context, the cryptography may not add to the security of the system. The canonical example is the use of cryptography to secure communications between two low-security systems. If only trusted users can access the two systems, cryptography protects messages in transit. But if untrusted users can access either system (through authorized accounts or, more likely, by breaking in), the cryptography is not sufficient to protect the messages. The attackers can read the messages at either endpoint.

The third goal is to demonstrate that computer security is not just a science but also an art. It is an art because no system can be considered secure without an examination of how it is to be used. The definition of a “secure computer” necessitates a statement of requirements and an expression of those requirements in the form of authorized actions and authorized users. (A computer engaged in work at a university may be considered “secure” for the purposes of the work done at the university. When moved to a military installation, that same system may not provide sufficient control to be deemed “secure” for the purposes of the work done at that installation.) How will people, as well as other computers, interact with the computer system? How clear and restrictive an interface can a designer create without rendering the system unusable while trying to prevent unauthorized use or access to the data or resources on the system?

Just as an artist paints his view of the world onto canvas, so does a designer of security features articulate his view of the world of human/machine interaction in the security policy and mechanisms of the system. Two designers may use entirely different designs to achieve the same creation, just as two artists may use different subjects to achieve the same concept.

Computer security is also a science. Its theory is based on mathematical constructions, analyses, and proofs. Its systems are built in accordance with the accepted practices of engineering. It uses inductive and deductive reasoning to examine the security of systems from key axioms and to discover underlying principles. These scientific principles can then be applied to untraditional situations and new theories, policies, and mechanisms.

Philosophy

Key to understanding the problems that exist in computer security is a recognition that the problems are not new. They are old problems, dating from the beginning of computer security (and, in fact, arising from parallel problems in the non-computer world). But the locus has changed as the field of computing has

changed. Before the mid-1980s, mainframe and mid-level computers dominated the market, and computer security problems and solutions were phrased in terms of securing files or processes on a single system. With the rise of networking and the Internet, the arena has changed. Workstations and servers, and the networking infrastructure that connects them, now dominate the market. Computer security problems and solutions now focus on a networked environment. However, if the workstations and servers, and the supporting network infrastructure, are viewed as a single system, the models, theories, and problem statements developed for systems before the mid-1980s apply equally well to current systems.

As an example, consider the issue of assurance. In the early period, assurance arose in several ways: formal methods and proofs of correctness, validation of policy to requirements, and acquisition of data and programs from trusted sources, to name a few. Those providing assurance analyzed a single system, the code on it, and the sources (vendors and users) from which the code could be acquired to ensure that either the sources could be trusted or the programs could be confined adequately to do minimal damage. In the later period, the same basic principles and techniques apply, except that the scope of some has been greatly expanded (from a single system and a small set of vendors to the world-wide Internet). The work on proof-carrying code, an exciting development in which the proof that a downloadable program module satisfies a stated policy is incorporated into the program itself, is an example of this expansion.³ It extends the notion of a proof of consistency with a stated policy. It advances the technology of the earlier period into the later period. But in order to understand it properly, one must understand the ideas underlying the concept of proof-carrying code, and these ideas lie in the earlier period.

As another example, consider Saltzer and Schroeder's principles of secure design.⁴ Enunciated in 1975, they promote simplicity, confinement, and understanding. When security mechanisms grow too complex, attackers can evade or bypass them. Many programmers and vendors are learning this when attackers break into their systems and servers. The argument that the principles are old, and somehow outdated, rings hollow when the result of their violation is a non-secure system.

The work from the earlier period is sometimes cast in terms of systems that no longer exist and that differ in many ways from modern systems. This does not vitiate the ideas and concepts, which also underlie the work done today. Once these ideas and concepts are properly understood, applying them in a multiplicity of environments becomes possible. Furthermore, the current mechanisms and technologies will become obsolete and of historical interest themselves as new forms of computing arise, but the underlying principles will live on, to underlie the next generation—indeed the next era—of computing.

The philosophy of this book is that certain key concepts underlie all of computer security, and that the study of all parts of computer security enriches

³Section 23.9.5.1 discusses proof-carrying code.

⁴Chapter 14 discusses these principles.

the understanding of all parts. Moreover, critical to an understanding of the applications of security-related technologies and methodologies is an understanding of the theory underlying those applications. Advances in the theory of computer protection have illuminated the foundations of security systems. Issues of abstract modeling, and modeling to meet specific environments, lead to systems designed to achieve a specific and rewarding goal. Theorems about composability of policies⁵ and the undecidability of the general security question⁶ have indicated the limits of what can be done. Much work and effort are continuing to extend the borders of those limits.

Application of these results has improved the quality of the security of the systems being protected. However, the issue is how compatibly the assumptions of the model (and theory) conform to the environment to which the theory is applied. Although our knowledge of how to apply these abstractions is continually increasing, we still have difficulty correctly transposing the relevant information from a realistic setting to one in which analyses can then proceed. Such abstraction often eliminates vital information. The omitted data may pertain to security in non-obvious ways. Without this information, the analysis is flawed.

The practitioner needs to know both the theoretical and practical aspects of the art and science of computer security. The theory demonstrates what is possible. The practical makes known what is feasible. The theoretician needs to understand the constraints under which these theories are used, how their results are translated into practical tools and methods, and how realistic are the assumptions underlying the theories. *Computer Security: Art and Science* tries to meet these needs.

Unfortunately, no single work can cover all aspects of computer security, so this book focuses on those parts that are, in the author's opinion, most fundamental and most pervasive. The mechanisms exemplify the applications of these principles.

Organization

The organization of this book reflects its philosophy. It begins with mathematical fundamentals and principles that provide boundaries within which security can be modeled and analyzed effectively. The mathematics provides a framework for expressing and analyzing the requirements of the security of a system. These policies constrain what is allowed and what is not allowed. Mechanisms provide the ability to implement these policies. The degree to which the mechanisms correctly implement the policies, and indeed the degree to which the policies themselves meet the requirements of the organizations using the system, are questions of assurance. Exploiting failures in policy, in implementation, and in assurance comes next, as well as mechanisms for providing information on the attack. The book concludes with the applications of both theory and policy focused

⁵See Chapter 9, "Noninterference and Policy Composition."

⁶See Section 3.2, "Basic Results."

on realistic situations. This natural progression emphasizes the development and application of the principles existent in computer security.

Part I, “Introduction,” describes what computer security is all about and explores the problems and challenges to be faced. It sets the context for the remainder of the book.

Part II, “Foundations,” deals with basic questions such as how “security” can be clearly and functionally defined, whether or not it is realistic, and whether or not it is decidable. If it is decidable, under what conditions is it decidable, and if not, how must the definition be bounded in order to make it decidable?

Part III, “Policy,” probes the relationship between policy and security. The definition of “security” depends on policy. In Part III we examine several types of policies, including the ever-present fundamental questions of trust, analysis of policies, and the use of policies to constrain operations and transitions.

Part IV, “Implementation I: Cryptography,” discusses cryptography and its role in security. It focuses on applications and discusses issues such as key management and escrow, key distribution, and how cryptosystems are used in networks. A quick study of authentication completes Part III.

Part V, “Implementation II: Systems,” considers how to implement the requirements imposed by policies using system-oriented techniques. Certain design principles are fundamental to effective security mechanisms. Policies define who can act and how they can act, and so identity is a critical aspect of implementation. Mechanisms implementing access control and flow control enforce various aspects of policies.

Part VI, “Assurance,” presents methodologies and technologies for ascertaining how well a system, or a product, meets its goals. After setting the background, to explain exactly what “assurance” is, the art of building systems to meet varying levels of assurance is discussed. Formal verification methods play a role. Part VI shows how the progression of standards has enhanced our understanding of assurance techniques.

Part VII, “Special Topics,” discusses some miscellaneous aspects of computer security. Malicious logic thwarts many mechanisms. Despite our best efforts at high assurance, systems today are replete with vulnerabilities. Why? How can a system be analyzed to detect vulnerabilities? What models might help us improve the state of the art? Given these security holes, how can we detect attackers who exploit them? A discussion of auditing flows naturally into a discussion of intrusion detection—a detection method for such attacks.

Part VIII, “Practicum,” presents examples of how to apply the principles discussed throughout the book. It begins with networks and proceeds to systems, users, and programs. Each chapter states a desired policy and shows how to translate that policy into a set of mechanisms and procedures that support the policy. Part VIII tries to demonstrate that the material covered elsewhere can be, and should be, used in practice.

Each chapter in this book ends with a summary, descriptions of some research issues, and some suggestions for further reading. The summary highlights the important ideas in the chapter. The research issues are current “hot topics” or are topics that may prove to be fertile ground for advancing the state of the art and

science of computer security. Interested readers who wish to pursue the topics in any chapter in more depth can go to some of the suggested readings. They expand on the material in the chapter or present other interesting avenues.

Roadmap

This book is both a reference book and a textbook. Its audience is undergraduate and graduate students as well as practitioners. This section offers some suggestions on approaching the book.

Dependencies

Chapter 1 is fundamental to the rest of the book and should be read first. After that, however, the reader need not follow the chapters in order. Some of the dependencies among chapters are as follows.

Chapter 3 depends on Chapter 2 and requires a fair degree of mathematical maturity. Chapter 2, on the other hand, does not. The material in Chapter 3 is for the most part not used elsewhere (although the existence of the first section's key result, the undecidability theorem, is mentioned repeatedly). It can be safely skipped if the interests of the reader lie elsewhere.

The chapters in Part III build on one another. The formalisms in Chapter 5 are called on in Chapters 20 and 21, but nowhere else. Unless the reader intends to delve into the sections on theorem proving and formal mappings, the formalisms may be skipped. The material in Chapter 9 requires a degree of mathematical maturity, and this material is used sparingly elsewhere. Like Chapter 3, Chapter 9 can be skipped by the reader whose interests lie elsewhere.

Chapters 10, 11, and 12 also build on one another in order. A reader who has encountered basic cryptography will have an easier time with the material than one who has not, but the chapters do not demand the level of mathematical experience that Chapters 3 and 9 require. Chapter 13 does not require material from Chapter 11 or Chapter 12, but it does require material from Chapter 10.

Chapter 14 is required for all of Part V. A reader who has studied operating systems at the undergraduate level will have no trouble with Chapter 16. Chapter 15 uses the material in Chapters 10 and 11; Chapter 17 builds on material in Chapters 5, 14, and 16; and Chapter 18 uses material in Chapters 4, 14, and 17.

Chapter 19 relies on information in Chapter 4. Chapter 20 builds on Chapters 5, 14, 16, and 19. Chapter 21 presents highly mathematical concepts and uses material from Chapters 19 and 20. Chapter 22 is based on material in Chapters 5, 19, and 20; it does not require Chapter 21. For all of Part VI, a knowledge of software engineering is very helpful.

Chapter 23 draws on ideas and information in Chapters 5, 6, 10, 14, 16, and 18 (and for Section 23.8, the reader should read Section 3.1). Chapter 24 is self-contained, although it implicitly uses many ideas from assurance. It also assumes a good working knowledge of compilers, operating systems, and in some cases networks. Many of the flaws are drawn from versions of the UNIX operating

system, or from Windows systems, and so a working knowledge of either or both systems will make some of the material easier to understand. Chapter 25 uses information from Chapter 4, and Chapter 26 uses material from Chapter 25.

The practicum chapters are self-contained and do not require any material beyond Chapter 1. However, they point out relevant material in other sections that augments the information and (we hope) the reader's understanding of that information.

Background

The material in this book is at the advanced undergraduate level. Throughout, we assume that the reader is familiar with the basics of compilers and computer architecture (such as the use of the program stack) and operating systems. The reader should also be comfortable with modular arithmetic (for the material on cryptography). Some material, such as that on formal methods (Chapter 21) and the mathematical theory of computer security (Chapter 3 and the formal presentation of policy models), requires considerable mathematical maturity. Other specific recommended background is presented in the preceding section. Part IX, the appendices, contains material that will be helpful to readers with backgrounds that lack some of the recommended material.

Examples are drawn from many systems. Many come from the UNIX operating system or variations of it (such as Linux). Others come from the Windows family of systems. Familiarity with these systems will help the reader understand many examples easily and quickly.

Undergraduate Level

An undergraduate class typically focuses on applications of theory and how students can use the material. The specific arrangement and selection of material depends on the focus of the class, but all classes should cover some basic material—notably that in Chapters 1, 10, and 14, as well as the notion of an access control matrix, which is discussed in Sections 2.1 and 2.2.

Presentation of real problems and solutions often engages undergraduate students more effectively than presentation of abstractions. The special topics and the practicum provide a wealth of practical problems and ways to deal with them. This leads naturally to the deeper issues of policy, cryptography, non-cryptographic mechanisms, and assurance. The following are sections appropriate for non-mathematical undergraduate courses in these topics.

- *Policy*: Sections 4.1 through 4.4 describe the notion of policy. The instructor should select one or two examples from Sections 5.1, 5.2.1, 6.2, 6.4, 8.1.1, and 8.2, which describe several policy models informally. Section 8.4 discusses role-based access control.
- *Cryptography*: Key distribution is discussed in Sections 11.1 and 11.2, and a common form of public key infrastructures (called PKIs) is discussed in Section 11.4.2. Section 12.1 points out common errors

in using cryptography. Section 12.4 shows how cryptography is used in networks, and the instructor should use one of the protocols in Section 12.5 as an example. Chapter 13 offers a look at various forms of authentication, including non-cryptographic methods.

- *Non-cryptographic mechanisms*: Identity is the basis for many access control mechanisms. Sections 15.1 through 15.4 discuss identity on a system, and Section 15.6 discusses identity and anonymity on the Web. Sections 16.1 and 16.2 explore two mechanisms for controlling access to files, and Section 16.4 discusses the ring-based mechanism underlying the notion of multiple levels of privilege. If desired, the instructor can cover sandboxes by using Sections 18.1 and 18.2, but because Section 18.2 uses material from Section 4.5, the instructor will need to go over those sections as well.
- *Assurance*: Chapter 19 provides a basic introduction to the often overlooked topic of assurance.

Graduate Level

A typical introductory graduate class can focus more deeply on the subject than can an undergraduate class. Like an undergraduate class, a graduate class should cover Chapters 1, 10, and 14. Also important are the undecidability results in Sections 3.1 and 3.2, which require that Chapter 2 be covered. Beyond that, the instructor can choose from a variety of topics and present them to whatever depth is appropriate. The following are sections suitable for graduate study.

- *Policy models*: Part III covers many common policy models both informally and formally. The formal description is much easier to understand once the informal description is understood, so in all cases both should be covered. The controversy in Section 5.4 is particularly illuminating to students who have not considered the role of policy and the nature of a policy. Chapter 9 is a highly formal discussion of the foundations of policy and is appropriate for students with experience in formal mathematics. Students without such a background will find it quite difficult.
- *Cryptography*: Part IV focuses on the applications of cryptography, not on cryptography's mathematical underpinnings.⁷ It discusses areas of interest critical to the use of cryptography, such as key management and some basic cryptographic protocols used in networking.
- *Non-cryptographic mechanisms*: Issues of identity and certification are complex and generally poorly understood. Section 15.5 covers these problems. Combining this with the discussion of identity on the Web

⁷The interested reader will find a number of books covering aspects of this subject [440, 787, 788, 914, 1092, 1093, 1318, 1826].

- (Section 15.6) raises issues of trust and naming. Chapters 17 and 18 explore issues of information flow and confining that flow.
- *Assurance*: Traditionally, assurance is taught as formal methods, and Chapter 21 serves this purpose. In practice, however, assurance is more often accomplished by using structured processes and techniques and informal but rigorous arguments of justification, mappings, and analysis. Chapter 20 emphasizes these topics. Chapter 22 discusses evaluation standards and relies heavily on the material in Chapters 19 and 20 and some of the ideas in Chapter 21.
 - *Miscellaneous Topics*: Section 23.8 presents a proof that the generic problem of determining if a generic program is a computer virus is in fact undecidable. The theory of penetration studies in Section 24.2, and the more formal approach in Section 24.6, illuminate the analysis of systems for vulnerabilities. If the instructor chooses to cover intrusion detection (Chapter 26) in depth, it should be understood that this discussion draws heavily on the material on auditing (Chapter 25).
 - *Practicum*: The practicum (Part VIII) ties the material in the earlier part of the book to real-world examples and emphasizes the applications of the theory and methodologies discussed earlier.

Practitioners

Practitioners in the field of computer security will find much to interest them. The table of contents and the index will help them locate specific topics. A more general approach is to start with Chapter 1 and then proceed to Part VIII, the practicum. Each chapter has references to other sections of the text that explain the underpinnings of the material. This will lead the reader to a deeper understanding of the reasons for the policies, settings, configurations, and advice in the practicum. This approach also allows readers to focus on those topics that are of most interest to them.

Register your copy of *Computer Security, Second Edition*, on the InformIT site for convenient access to updates and/or corrections as they become available. To start the registration process, go to informit.com/register and log in or create an account. Enter the product ISBN (9780321712332) and click Submit. Look on the Registered Products tab for an Access Bonus Content link next to this product, and follow that link to access any available bonus materials. If you would like to be notified of exclusive offers on new editions and updates, please check the box to receive email from us.

This page intentionally left blank

Acknowledgments

It is not possible to separate those who contributed to the second edition from those who contributed to the first edition, because everything done for the first edition, especially after the first printing, has contributed to the second. So these acknowledgments apply to both editions. That said . . .

Special Acknowledgments

Elisabeth Sullivan and Michelle Ruppel contributed the assurance part of this book.

For the first edition, Liz wrote several drafts, all of which reflect her extensive knowledge and experience in that aspect of computer security. I am particularly grateful to her for contributing her real-world knowledge of how assurance is managed. Too often, books recount the mathematics of assurance without recognizing that other aspects are equally important and more widely used. These other aspects shine through in the assurance section, thanks to Liz. As if that were not enough, she made several suggestions that improved the policy part of this book. I will always be grateful for her contribution, her humor, and especially her friendship.

For the second edition, Michelle stepped in to update that part based on her extensive experience and real-world knowledge as a practitioner. She was careful to maintain the tone and style of Liz's writing, and her contributions strengthened the assurance part. I am grateful to her for agreeing to step in, for the exceptional effort she put forth, and the high quality that resulted.

In summary, I am very grateful for their contributions.

Acknowledgments

Many people offered comments, suggestions, and ideas for the second edition. Thanks to Marvin Schaefer, Sean Peisert, Prof. Christian Probst, Carrie Gates, and Richard Ford for their reviews of the various chapters. I appreciate Prof. Ken

Rosen and Prof. Alfred Menezes for their help with Chapter 10, Steven Templeton and Kara Nance for their suggestions on Chapter 27, Karl Levitt for his comments on Chapter 26, and Richard Ford for his many suggestions on Chapter 23. Their advice and suggestions were invaluable in preparing this edition. Of course, any errors in the text are my responsibility, and usually occurred because I did not always follow their advice.

Thanks also to Pasquale Noce, who sent me a thorough analysis of many of the theorems, proving them constructively as opposed to how they were done in the book. He made many other helpful comments and caught some errors.

The students in Peter Reiher's COM SCI 236-80, Computer Security, class at UCLA in the Spring Quarter 2018, and the students in my ECS 153, Computer Security, classes over the past few years at UC Davis used parts of this edition in various stages of preparation. I thank them for their feedback, which also improved the book.

Many others contributed to this book in various ways. Special thanks to Steven Alexander, Amin Alipour, Jim Alves-Foss, Bill Arbaugh, Andrew Arcilla, Alex Aris, Rebecca Bace, Belinda Bashore, Vladimir Berman, Rafael Bhatti, Ziad El Bizri, David Bover, Logan Browne, Terry Brugger, Gordon Burns, Serdar Cabuk, Raymond Centeno, Yang Chen, Yi Chen, HakSoo Choi, Lisa Clark, Michael Clifford, Christopher Clifton, Dan Coming, Kay Connelly, Crispin Cowan, Shayne Czyzewski, Tom Daniels, Dimitri DeFigueiredo, Joseph-Patrick Dib, Till Döriges, Felix Fan, Robert Fourny, Guillermo Francia III, Jeremy Frank, Conny Francke, Martin Gagne, Nina Gholami, Ron Gove, James Hinde, James Hook, Xuxian Jiang, Jesper Johansson, Mark Jones, Calvin Ko, Mark-Neil Ledesma, Ken Levine, Karl Levitt, Luc Longpre, Yunhua Lu, Gary McGraw, Alexander Meau, Nasir Memon, Katherine Moore, Mark Morrissey, Ather Nawaz, Iulian Neamtui, Dan Nerenburg, Kimberly Nico, Stephen Northcutt, Rafael Obelheiro, Josko Orsulic, Holly Pang, Sean Peisert, Ryan Poling, Sung Park, Ashwini Raina, Jorge Ramos, Brennen Reynolds, Peter Rozentel, Christoph Schuba, night SH, David Shambroom, Jonathan Shapiro, Clay Shields, Sriram Srinivasan, Mahesh V. Tripunitara, Vinay Vittal, Tom Walcott, James Walden, Dan Watson, Guido Wedig, Chris Wee, Han Weili, Patrick Wheeler, Paul Williams, Bonnie Xu, Charles Yang, Xiaoduan Ye, Xiaohui Ye, Lara Whelan, John Zachary, Linfeng Zhang, Aleksandr Zingorenko, and to everyone in my and others' computer security classes, who (knowingly or unknowingly) helped me develop and test this material.

The Pearson folks, in particular my editors Laura Lewin and Malobika Chakraborty, and Sheri Replin, were incredibly helpful and patient. Their patience and enthusiasm ensured this second edition was completed, although a bit later than expected. The production people, especially Julie Nahil, Ramya Gangadharan, and Charles Roumeliotis, moved the book smoothly into print, and I thank them for making it as painless as possible. I owe them many thanks. Similarly, for the first edition, the Addison-Wesley folks, Kathleen Billus, Susannah Buzard, Bernie Gaffney, Amy Fleischer, Helen Goldstein, Tom Stone, Asdis Thorsteins-son, and most especially my editor, Peter Gordon, were incredibly patient and

helpful, despite fears that this book would never materialize. The fact that it did so is in great measure attributable to their hard work and encouragement. I also thank the production people at Argosy, especially Beatriz Valdés and Craig Kirkpatrick, for their wonderful work.

Dorothy Denning, my advisor in graduate school, guided me through the maze of computer security when I was just beginning. Peter Denning, Barry Leiner, Karl Levitt, Peter Neumann, Marvin Schaefer, Larry Snyder, and several others influenced my approach to the subject. I hope this work reflects in some small way what they gave to me and passes a modicum of it along to my readers.

I also thank my parents, Leonard Bishop and Linda Allen. My father, a writer, gave me some useful tips on writing, which I tried to follow. My mother, a literary agent, helped me understand the process of getting the book published, and supported me throughout.

Finally, I would like to thank my family for their support throughout the writing. My wife Holly, our children Heidi, Steven, David, and Caroline, and grandchildren Skyler and Sage were very patient and understanding and made sure I had time to work on the book. They also provided delightful distractions. To them all, my love and gratitude.

This page intentionally left blank

About the Author



Matt Bishop is a professor in the Department of Computer Science at the University of California at Davis. He received his Ph.D. in computer science from Purdue University, where he specialized in computer security, in 1984. He was a systems programmer at Megatest Corporation, a research scientist at the Research Institute of Advanced Computer Science and was on the faculty at Dartmouth College.

His main research area is the analysis of vulnerabilities in computer systems, including modeling them, building tools to detect vulnerabilities, and ameliorating or eliminating them. This includes detecting and handling all types of malicious logic. He works in the areas of network security, the study of denial of service attacks and defenses, policy modeling, software assurance testing, resilience, and formal modeling of access control. He has worked extensively in electronic voting, was one of the members of the RABA study for Maryland, and was one of the two principle investigators of the California Top-to-Bottom Review, which performed a technical review of all electronic voting systems certified in the State of California.

He is active in information assurance education. He was co-chair of the Joint Task Force that developed the *Cybersecurity Curricula 2017: Curriculum Guidelines for Post-Secondary Degree Programs in Cybersecurity*, released in December 2017. He teaches introductory programming, software engineering, operating systems, and (of course) computer security.

This page intentionally left blank

Part I

Introduction

Writers say “To write a good book, tell them what you are going to tell them, then tell them, then tell them what you told them.” This is the “what we’re going to tell you” part.

Chapter 1, “An Overview of Computer Security,” presents the underpinnings of computer security and an overview of the important issues to place them in context. It begins with a discussion of what computer security is and how threats are connected to security services. The combination of desired services makes up a policy, and mechanisms enforce the policy. All rely on underlying assumptions, and the systems built on top of these assumptions lead to issues of assurance. Finally, the operational and human factors affect the mechanisms used as well as the policy.

This page intentionally left blank

Chapter 1

An Overview of Computer Security

ANTONIO: Whereof what's past is prologue, what to come
In yours and my discharge.
— *The Tempest*, II, i, 257–258.

This chapter presents the basic concepts of computer security. The remainder of this book will elaborate on these concepts in order to reveal the logic underlying the principles of these concepts.

We begin with basic security-related services that protect against threats to the security of the system. The next section discusses security policies that identify the threats and define the requirements for ensuring a secure system. Security mechanisms detect and prevent attacks and recover from those that succeed. Analyzing the security of a system requires an understanding of the mechanisms that enforce the security policy. It also requires a knowledge of the related assumptions and trust, which leads to the threats and the degree to which they may be realized. Such knowledge allows one to design better mechanisms and policies to neutralize these threats. This process leads to risk analysis. Human beings are the weakest link in the security mechanisms of any system. Therefore, policies and procedures must take people into account. This chapter discusses each of these topics.

1.1 The Basic Components

Computer security rests on confidentiality, integrity, and availability. The interpretations of these three aspects vary, as do the contexts in which they arise. The interpretation of an aspect in a given environment is dictated by the needs of the individuals, customs, and laws of the particular organization.

1.1.1 Confidentiality

Confidentiality is the concealment of information or resources. The need for keeping information secret arises from the use of computers in institutions with sensitive information such as government and industry. For example, military and civilian institutions in the government often restrict access to information to those who need that information. The first formal work in computer security was motivated by the military's attempt to implement controls to enforce a "need to know" principle. This principle also applies to industrial firms, which keep their proprietary designs secure lest their competitors try to steal the designs. As a further example, all types of institutions keep some types of personnel records secret.

Access control mechanisms support confidentiality. One access control mechanism for preserving confidentiality is *cryptology*, which transforms data to make it incomprehensible. A *cryptographic key* controls access to the untransformed data, but then the cryptographic key itself becomes another datum to be protected.

EXAMPLE: Enciphering an income tax return will prevent anyone without the key from reading the taxable income on the return. If the owner needs to see the return, it must be deciphered. Only the possessor of the cryptographic key can enter it into a deciphering program. However, if someone else can read the key when it is entered into the program and has access to the enciphered return, the confidentiality of the tax return has been compromised.

Other system-dependent mechanisms can prevent information from being illicitly accessed. Data protected only by these controls can be read when the controls fail or are bypassed. Then the controls' advantage is offset by a corresponding disadvantage. They can protect the secrecy of data more completely than cryptography, but if they fail or are evaded, the data becomes visible.

Confidentiality also applies to the existence of data, which is sometimes more revealing than the data itself. The precise number of people who distrust a politician may be less important than knowing that such a poll was taken by the politician's staff. How a particular government agency harassed citizens in its country may be less important than knowing that such harassment occurred. Access control mechanisms sometimes conceal the mere existence of data, lest the existence itself reveal information that should be protected.

Resource hiding is another important aspect of confidentiality. Organizations often wish to conceal their network configuration as well as what systems they are using. They may not wish others to know about specific equipment (because it could be used without authorization or in inappropriate ways), and a company renting time from a service provider may not want others to know what resources it is using. Access control mechanisms provide these capabilities as well.

All the mechanisms that enforce confidentiality require supporting services from the system. The assumption is that the security services can rely on the kernel,

and other agents, to supply correct data. Thus, assumptions and trust underlie confidentiality mechanisms.

1.1.2 Integrity

Integrity refers to the trustworthiness of data or resources, and it is usually phrased in terms of preventing improper or unauthorized change. Integrity includes data integrity (the content of the information) and origin integrity (the source of the data, often called *authentication*). The source of the information may bear on its accuracy and credibility and on the trust that people place in the information. This dichotomy illustrates the principle that the aspect of integrity known as credibility is central to the proper functioning of a system. We will return to this issue when discussing malicious logic.

EXAMPLE: A newspaper may print information obtained from a leak at the White House but attribute it to the wrong source. The information is printed as received (preserving data integrity), but its source is incorrect (corrupting origin integrity).

Integrity mechanisms fall into two classes: *prevention* mechanisms and *detection* mechanisms.

Prevention mechanisms seek to maintain the integrity of the data by blocking any unauthorized attempts to change the data or any attempts to change the data in unauthorized ways. The distinction between these two types of attempts is important. The former occurs when a user tries to change data that she has no authority to change. The latter occurs when a user authorized to make certain changes in the data tries to change the data in other ways. For example, suppose an accounting system is on a computer. Someone breaks into the system and tries to modify the accounting data. Here an unauthorized user has tried to violate the integrity of the accounting database. But if an accountant hired by the firm to maintain its books tries to embezzle money by sending it overseas and hiding the transactions, a user (the accountant) has tried to change data (the accounting data) in unauthorized ways (by not entering the transfer of funds to a Swiss bank account). Adequate authentication and access controls will generally stop the break-in from the outside, but preventing the second type of attempt requires very different controls.

Detection mechanisms do not try to prevent violations of integrity; they simply report that the data's integrity is no longer trustworthy. Detection mechanisms may analyze system events (user or system actions) to detect problems or (more commonly) may analyze the data itself to see if required or expected constraints still hold. The mechanisms may report the actual cause of the integrity violation (a specific part of a file was altered), or they may simply report that the file is now corrupt.

Working with integrity is very different than working with confidentiality. With confidentiality, the data is either compromised or it is not, but integrity includes both the correctness and the trustworthiness of the data. The origin of the data (how and from whom it was obtained), how well the data was protected

before it arrived at the current machine, and how well the data is protected on the current machine all affect the integrity of the data. Thus, evaluating integrity is often very difficult, because it relies on assumptions about the source of the data and about trust in that source—two underpinnings of security that are often overlooked.

1.1.3 Availability

Availability refers to the ability to use information or resources. Availability is an important aspect of reliability as well as of system design because an unavailable system is at least as bad as no system at all. The aspect of availability that is relevant to security is that someone may deliberately arrange to deny access to data or to a service by making it unavailable or unusable. System designs usually assume a statistical model to analyze expected patterns of use, and mechanisms ensure availability when that statistical model holds. Someone may be able to manipulate use (or parameters that control use, such as network traffic) so that the assumptions of the statistical model are no longer valid. This means that the mechanisms for keeping the resource or data available are working in an environment for which they were not designed. As a result, they will often fail.

EXAMPLE: Suppose Anne has compromised a bank's secondary system server, which supplies bank account balances. When anyone else asks that server for information, Anne can supply any information she desires. Merchants validate checks by contacting the bank's primary balance server. If a merchant gets no response, the secondary server will be asked to supply the data. Anne's colleague prevents merchants from contacting the primary balance server, so all merchant queries go to the secondary server. Anne will never have a check turned down, regardless of her actual account balance. Notice that if the bank had only one server (the primary one) and that server were unavailable, this scheme would not work. The merchant would be unable to validate the check.

Attempts to block availability, called *denial of service (DoS) attacks*, can be the most difficult to detect, because the analyst must determine if the unusual access patterns are attributable to deliberate manipulation of resources or of environment. Complicating this determination is the nature of statistical models. Even if the model accurately describes the environment, atypical events simply contribute to the nature of the statistics. A deliberate attempt to make a resource unavailable may look like, or be, an atypical event. In some environments, it may not even appear atypical.

1.2 Threats

A *threat* is a potential violation of security. The violation need not actually occur for there to be a threat. The fact that the violation *might* occur means that those

actions that could cause it to occur must be guarded against (or prepared for). Those actions are called *attacks*. Those who execute such actions, or cause them to be executed, are called *attackers*.

The three security services—confidentiality, integrity, and availability—counter threats to the security of a system. Shirey [1739] divides threats into four broad classes: *disclosure*, or unauthorized access to information; *deception*, or acceptance of false data; *disruption*, or interruption or prevention of correct operation; and *usurpation*, or unauthorized control of some part of a system. These four broad classes encompass many common threats. Because the threats are ubiquitous, an introductory discussion of each one will present issues that recur throughout the study of computer security.

Snooping or *eavesdropping*, the unauthorized interception of information, is a form of disclosure. It is passive, suggesting simply that some entity is listening to (or reading) communications or browsing through files or system information. *Passive wiretapping* is a form of snooping in which a network is monitored. (It is called “wiretapping” because of the “wires” that compose the network, although the term is used even if no physical wiring is involved.) Confidentiality services seek to counter this threat.

Modification or *alteration*, an unauthorized change of information, covers three classes of threats. The goal may be deception, in which some entity relies on the modified data to determine which action to take, or in which incorrect information is accepted as correct and is released. If the modified data controls the operation of the system, the threats of disruption and usurpation arise. Unlike snooping, modification is active; it results from an entity changing information. *Active wiretapping* is a form of modification in which data moving across a network is altered, new data is injected, or parts of the data are deleted; the term “active” distinguishes it from snooping (“passive” wiretapping). An example is the *man-in-the-middle* attack, in which an intruder reads messages from the sender and sends (possibly modified) versions to the recipient, in hopes that the recipient and sender will not realize the presence of the intermediary. Integrity services seek to counter this threat.

Masquerading or *spoofing*, an impersonation of one entity by another, is a form of both deception and usurpation. It lures a victim into believing that the entity with which it is communicating is a different entity. For example, if a user tries to log into a computer across the Internet but instead reaches another computer that claims to be the desired one, the user has been spoofed. Similarly, if a user tries to read a web page, but an attacker has arranged for the user to be given a different page, another spoof has taken place. This may be a passive attack (in which the user simply accesses the web page), but it is usually an active attack (in which the attacker issues responses dynamically to mislead the user about the web page). Although masquerading is primarily deception, it is often used to usurp control of a system by an attacker impersonating an authorized manager or controller. Integrity services (called “authentication services” in this context) seek to counter this threat.

Some forms of masquerading may be allowed. *Delegation* occurs when one entity authorizes a second entity to perform functions on its behalf. The distinctions between delegation and masquerading are important. If Susan delegates to

Thomas the authority to act on her behalf, she is giving permission for him to perform specific actions as though she were performing them herself. All parties are aware of the delegation. Thomas will not pretend to be Susan; rather, he will say, “I am Thomas and I have authority to do this on Susan’s behalf.” If asked, Susan will verify this. On the other hand, in a masquerade, Thomas will pretend to be Susan. No other parties (including Susan) will be aware of the masquerade, and Thomas will say, “I am Susan.” Should anyone discover that he or she is dealing with Thomas and ask Susan about it, she will deny that she authorized Thomas to act on her behalf. Even though masquerading is a violation of security, delegation is not.

Repudiation of origin, a false denial that an entity sent (or created) something, is a form of deception. For example, suppose a customer sends a letter to a vendor agreeing to pay a large amount of money for a product. The vendor ships the product and then demands payment. The customer denies having ordered the product and, according to a law in the customer’s state, is therefore entitled to keep the unsolicited shipment without payment. The customer has repudiated the origin of the letter. If the vendor cannot prove that the letter came from the customer, the attack succeeds. A variant of this is denial by a user that he created specific information or entities such as files. Integrity mechanisms try to cope with this threat.

Denial of receipt, a false denial that an entity received some information or message, is a form of deception. Suppose a customer orders an expensive product, but the vendor demands payment before shipment. The customer pays, and the vendor ships the product. The customer then asks the vendor when he will receive the product. If the customer has already received the product, the question constitutes a denial of receipt attack. The vendor can defend against this attack only by proving that the customer did, despite his denials, receive the product. Integrity and availability mechanisms attempt to guard against these attacks.

Delay, a temporary inhibition of a service, is a form of usurpation, although it can play a supporting role in deception. Typically, delivery of a message or service requires some time t ; if an attacker can force the delivery to take more than time t , the attacker has successfully delayed delivery. This requires manipulation of system control structures, such as network components or server components, and hence is a form of usurpation. If an entity is waiting for an authorization message that is delayed, it may query a secondary server for the authorization. Even though the attacker may be unable to masquerade as the primary server, she might be able to masquerade as that secondary server and supply incorrect information. Availability mechanisms can often thwart this threat.

Denial of service, a long-term inhibition of service, is a form of usurpation, although it is often used with other mechanisms to deceive. The attacker prevents a server from providing a service. The denial may occur at the source (by preventing the server from obtaining the resources needed to perform its function), at the destination (by blocking the communications from the server), or along the intermediate path (by discarding messages from either the client or the server, or both). Denial of service poses the same threat as an infinite delay. Availability mechanisms seek to counter this threat.

Denial of service or delay may result from direct attacks or from problems unrelated to security. From our point of view, the cause and result are important; the intention underlying them is not. If delay or denial of service compromises system security, or is part of a sequence of events leading to the compromise of a system, then we view it as an attempt to breach system security. But the attempt may not be deliberate; indeed, it may be a user error, or the product of environmental characteristics, rather than specific actions of an attacker.

1.3 Policy and Mechanism

Critical to our study of security is the distinction between policy and mechanism:

Definition 1–1. A *security policy* is a statement of what is, and what is not, allowed.

Definition 1–2. A *security mechanism* is a method, tool, or procedure for enforcing a security policy.

Mechanisms can be nontechnical, such as requiring proof of identity before changing a password; in fact, policies often require some procedural mechanisms that technology cannot enforce.

As an example, suppose a university's computer science laboratory has a policy that prohibits any student from copying another student's homework files. The computer system provides mechanisms for preventing others from reading a user's files. Anna fails to use these mechanisms to protect her homework files, and Bill copies them. A breach of security has occurred, because Bill has violated the security policy. Anna's failure to protect her files does not authorize Bill to copy them.

In this example, Anna could easily have protected her files. In other environments, such protection may not be easy. For example, the Internet provides only the most rudimentary security mechanisms, which are not adequate to protect information sent over that network. Nevertheless, acts such as the recording of passwords and other sensitive information violate an implicit security policy of most sites (specifically, that passwords are a user's confidential property and cannot be recorded by anyone).

Policies may be presented mathematically, as a list of allowed (secure) and disallowed (nonsecure) states. For our purposes, we will assume that any given policy provides an axiomatic description of secure states and nonsecure states. In practice, policies are rarely so precise; they normally describe in English, or some other natural language, what users and staff are allowed to do. The ambiguity inherent in such a description leads to states that are not classified as "allowed" or "disallowed." For example, consider the homework policy discussed previously. If someone looks through another user's directory without copying homework