Foreword xiii

Preface xvii

1. Safe Initialization 1

1.1 Sanitizing the Environment 1
1.2 Restricting Privileges on Windows 7
1.3 Dropping Privileges in setuid Programs 16
1.4 Limiting Risk with Privilege Separation 20
1.5 Managing File Descriptors Safely 23
1.6 Creating a Child Process Securely 26
1.7 Executing External Programs Securely 28
1.8 Executing External Programs Securely 33
1.9 Disabling Memory Dumps in the Event of a Crash 35

2. Access Control 38

2.1 Understanding the Unix Access Control Model 38
2.2 Understanding the Windows Access Control Model 41
2.3 Determining Whether a User Has Access to a File on Unix 43
2.4 Determining Whether a Directory Is Secure 45
2.5 Erasing Files Securely 47
2.6 Accessing File Information Securely 53
2.7 Restricting Access Permissions for New Files on Unix 55
2.8 Locking Files 57
2.9 Synchronizing Resource Access Across Processes on Unix 60
2.10 Synchronizing Resource Access Across Processes on Windows 63
2.11 Creating Files for Temporary Use 65
2.12 Restricting Filesystem Access on Unix 68
2.13 Restricting Filesystem and Network Access on FreeBSD 69

3. Input Validation **71**
3.1 Understanding Basic Data Validation Techniques 71
3.2 Preventing Attacks on Formatting Functions 75
3.3 Preventing Buffer Overflows 78
3.4 Using the SafeStr Library 85
3.5 Preventing Integer Coercion and Wrap-Around Problems 88
3.6 Using Environment Variables Securely 92
3.7 Validating Filenames and Paths 97
3.8 Evaluating URL Encodings 99
3.9 Validating Email Addresses 101
3.10 Preventing Cross-Site Scripting 103
3.11 Preventing SQL Injection Attacks 107
3.12 Detecting Illegal UTF-8 Characters 110
3.13 Preventing File Descriptor Overflows When Using select() 112

4. Symmetric Cryptography Fundamentals **116**
4.1 Representing Keys for Use in Cryptographic Algorithms 117
4.2 Generating Random Symmetric Keys 119
4.3 Representing Binary Keys (or Other Raw Data) as Hexadecimal 120
4.4 Turning ASCII Hex Keys (or Other ASCII Hex Data) into Binary 121
4.5 Performing Base64 Encoding 123
4.6 Performing Base64 Decoding 125
4.7 Representing Keys (or Other Binary Data) as English Text 128
4.8 Converting Text Keys to Binary Keys 130
4.9 Using Salts, Nonces, and Initialization Vectors 133
4.10 Deriving Symmetric Keys from a Password 136
4.11 Algorithmically Generating Symmetric Keys from One Base Secret 141
4.12 Encrypting in a Single Reduced Character Set 146
4.13 Managing Key Material Securely 149
4.14 Timing Cryptographic Primitives 150

5. Symmetric Encryption **155**
5.1 Deciding Whether to Use Multiple Encryption Algorithms 155
5.2 Figuring Out Which Encryption Algorithm Is Best 156
5.3 Selecting an Appropriate Key Length 160
5.4 Selecting a Cipher Mode 162
5.5 Using a Raw Block Cipher 171
5.6 Using a Generic CBC Mode Implementation 175
5.7 Using a Generic CFB Mode Implementation 186

5.8 Using a Generic OFB Mode Implementation 192
5.9 Using a Generic CTR Mode Implementation 197
5.10 Using CWC Mode 202
5.11 Manually Adding and Checking Cipher Padding 205
5.12 Precomputing Keystream in OFB, CTR, CCM, or CWC Modes (or with Stream Ciphers) 207
5.13 Parallelizing Encryption and Decryption in Modes That Allow It (Without Breaking Compatibility) 208
5.14 Parallelizing Encryption and Decryption in Arbitrary Modes (Breaking Compatibility) 212
5.15 Performing File or Disk Encryption 213
5.16 Using a High-Level, Error-Resistant Encryption and Decryption API 217
5.17 Performing Block Cipher Setup (for CBC, CFB, OFB, and ECB Modes) in OpenSSL 221
5.18 Using Variable Key-Length Ciphers in OpenSSL 226
5.19 Disabling Cipher Padding in OpenSSL in CBC Mode 227
5.20 Performing Additional Cipher Setup in OpenSSL 228
5.21 Querying Cipher Configuration Properties in OpenSSL 229
5.22 Performing Low-Level Encryption and Decryption with OpenSSL 230
5.23 Setting Up and Using RC4 233
5.24 Using One-Time Pads 236
5.25 Using Symmetric Encryption with Microsoft's CryptoAPI 237
5.26 Creating a CryptoAPI Key Object from Raw Key Data 244
5.27 Extracting Raw Key Data from a CryptoAPI Key Object 246

6. Hashes and Message Authentication 249

6.1 Understanding the Basics of Hashes and MACs 249
6.2 Deciding Whether to Support Multiple Message Digests or MACs 253
6.3 Choosing a Cryptographic Hash Algorithm 254
6.4 Choosing a Message Authentication Code 258
6.5 Incrementally Hashing Data 262
6.6 Hashing a Single String 267
6.7 Using a Cryptographic Hash 269
6.8 Using a Nonce to Protect Against Birthday Attacks 270
6.9 Checking Message Integrity 274
6.10 Using HMAC 276
6.11 Using OMAC (a Simple Block Cipher–Based MAC) 280
6.12 Using HMAC or OMAC with a Nonce 285
6.13 Using a MAC That's Reasonably Fast in Software and Hardware 286

6.14 Using a MAC That's Optimized for Software Speed 287
6.15 Constructing a Hash Function from a Block Cipher 291
6.16 Using a Block Cipher to Build a Full-Strength Hash Function 294
6.17 Using Smaller MAC Tags 298
6.18 Making Encryption and Message Integrity Work Together 298
6.19 Making Your Own MAC 300
6.20 Encrypting with a Hash Function 301
6.21 Securely Authenticating a MAC (Thwarting Capture Replay Attacks) 303
6.22 Parallelizing MACs 304

7. Public Key Cryptography 307
7.1 Determining When to Use Public Key Cryptography 309
7.2 Selecting a Public Key Algorithm 311
7.3 Selecting Public Key Sizes 312
7.4 Manipulating Big Numbers 315
7.5 Generating a Prime Number (Testing for Primality) 323
7.6 Generating an RSA Key Pair 327
7.7 Disentangling the Public and Private Keys in OpenSSL 329
7.8 Converting Binary Strings to Integers for Use with RSA 330
7.9 Converting Integers into Binary Strings for Use with RSA 331
7.10 Performing Raw Encryption with an RSA Public Key 332
7.11 Performing Raw Decryption Using an RSA Private Key 336
7.12 Signing Data Using an RSA Private Key 338
7.13 Verifying Signed Data Using an RSA Public Key 340
7.14 Securely Signing and Encrypting with RSA 343
7.15 Using the Digital Signature Algorithm (DSA) 347
7.16 Representing Public Keys and Certificates in Binary (DER Encoding) 352
7.17 Representing Keys and Certificates in Plaintext (PEM Encoding) 355

8. Authentication and Key Exchange 362
8.1 Choosing an Authentication Method 362
8.2 Getting User and Group Information on Unix 372
8.3 Getting User and Group Information on Windows 375
8.4 Restricting Access Based on Hostname or IP Address 379
8.5 Generating Random Passwords and Passphrases 387
8.6 Testing the Strength of Passwords 391
8.7 Prompting for a Password 392
8.8 Throttling Failed Authentication Attempts 398
8.9 Performing Password-Based Authentication with crypt( ) 400

8.10 Performing Password-Based Authentication with MD5-MCF 402
8.11 Performing Password-Based Authentication with PBKDF2 408
8.12 Authenticating with PAM 411
8.13 Authenticating with Kerberos 414
8.14 Authenticating with HTTP Cookies 419
8.15 Performing Password-Based Authentication and Key Exchange 422
8.16 Performing Authenticated Key Exchange Using RSA 429
8.17 Using Basic Diffie-Hellman Key Agreement 432
8.18 Using Diffie-Hellman and DSA Together 436
8.19 Minimizing the Window of Vulnerability When Authenticating Without a PKI 438
8.20 Providing Forward Secrecy in a Symmetric System 444
8.21 Ensuring Forward Secrecy in a Public Key System 445
8.22 Confirming Requests via Email 447

9. Networking 454
9.1 Creating an SSL Client 455
9.2 Creating an SSL Server 457
9.3 Using Session Caching to Make SSL Servers More Efficient 460
9.4 Securing Web Communication on Windows Using the WinInet API 463
9.5 Enabling SSL without Modifying Source Code 468
9.6 Using Kerberos Encryption 470
9.7 Performing Interprocess Communication Using Sockets 475
9.8 Performing Authentication with Unix Domain Sockets 482
9.9 Performing Session ID Management 486
9.10 Securing Database Connections 487
9.11 Using a Virtual Private Network to Secure Network Connections 490
9.12 Building an Authenticated Secure Channel Without SSL 491

10. Public Key Infrastructure 502
10.1 Understanding Public Key Infrastructure (PKI) 502
10.2 Obtaining a Certificate 513
10.3 Using Root Certificates 519
10.4 Understanding X.509 Certificate Verification Methodology 522
10.5 Performing X.509 Certificate Verification with OpenSSL 525
10.6 Performing X.509 Certificate Verification with CryptoAPI 530
10.7 Verifying an SSL Peer's Certificate 535
10.8 Adding Hostname Checking to Certificate Verification 539
10.9 Using a Whitelist to Verify Certificates 544

10.10 Obtaining Certificate Revocation Lists with OpenSSL 547
10.11 Obtaining CRLs with CryptoAPI 556
10.12 Checking Revocation Status via OCSP with OpenSSL 562

11. Random Numbers 568
11.1 Determining What Kind of Random Numbers to Use 568
11.2 Using a Generic API for Randomness and Entropy 573
11.3 Using the Standard Unix Randomness Infrastructure 575
11.4 Using the Standard Windows Randomness Infrastructure 580
11.5 Using an Application-Level Generator 581
11.6 Reseeding a Pseudo-Random Number Generator 591
11.7 Using an Entropy Gathering Daemon–Compatible Solution 594
11.8 Getting Entropy or Pseudo-Randomness Using EGADS 599
11.9 Using the OpenSSL Random Number API 603
11.10 Getting Random Integers 605
11.11 Getting a Random Integer in a Range 606
11.12 Getting a Random Floating-Point Value with Uniform Distribution 608
11.13 Getting Floating-Point Values with Nonuniform Distributions 609
11.14 Getting a Random Printable ASCII String 611
11.15 Shuffling Fairly 612
11.16 Compressing Data with Entropy into a Fixed-Size Seed 613
11.17 Getting Entropy at Startup 614
11.18 Statistically Testing Random Numbers 615
11.19 Performing Entropy Estimation and Management 621
11.20 Gathering Entropy from the Keyboard 630
11.21 Gathering Entropy from Mouse Events on Windows 638
11.22 Gathering Entropy from Thread Timings 643
11.23 Gathering Entropy from System State 644

12. Anti-Tampering 647
12.1 Understanding the Problem of Software Protection 648
12.2 Detecting Modification 653
12.3 Obfuscating Code 658
12.4 Performing Bit and Byte Obfuscation 664
12.5 Performing Constant Transforms on Variables 667
12.6 Merging Scalar Variables 667
12.7 Splitting Variables 669
12.8 Disguising Boolean Values 670
12.9 Using Function Pointers 671

12.10 Restructuring Arrays 672
12.11 Hiding Strings 678
12.12 Detecting Debuggers 681
12.13 Detecting Unix Debuggers 682
12.14 Detecting Windows Debuggers 685
12.15 Detecting SoftICE 685
12.16 Countering Disassembly 688
12.17 Using Self-Modifying Code 693

13. Other Topics 700
13.1 Performing Error Handling 700
13.2 Erasing Data from Memory Securely 704
13.3 Preventing Memory from Being Paged to Disk 707
13.4 Using Variable Arguments Properly 709
13.5 Performing Proper Signal Handling 712
13.6 Protecting against Shatter Attacks on Windows 716
13.7 Guarding Against Spawning Too Many Threads 718
13.8 Guarding Against Creating Too Many Network Sockets 724
13.9 Guarding Against Resource Starvation Attacks on Unix 727
13.10 Guarding Against Resource Starvation Attacks on Windows 730
13.11 Following Best Practices for Audit Logging 734

Index 739