

CONTENTS

PREFACE

xxiii

1 INTRODUCTION

1

- 1.1 WHAT IS AN OPERATING SYSTEM? 3
 - 1.1.1 The Operating System as an Extended Machine 4
 - 1.1.2 The Operating System as a Resource Manager 5

- 1.2 HISTORY OF OPERATING SYSTEMS 6
 - 1.2.1 The First Generation (1945–55): Vacuum Tubes 7
 - 1.2.2 The Second Generation (1955–65): Transistors and Batch Systems 8
 - 1.2.3 The Third Generation (1965–1980): ICs and Multiprogramming 9
 - 1.2.4 The Fourth Generation (1980–Present): Personal Computers 14
 - 1.2.5 The Fifth Generation (1990–Present): Mobile Computers 19

- 1.3 COMPUTER HARDWARE REVIEW 20
 - 1.3.1 Processors 21
 - 1.3.2 Memory 24
 - 1.3.3 Disks 27
 - 1.3.4 I/O Devices 28
 - 1.3.5 Buses 31
 - 1.3.6 Booting the Computer 34

- 1.4 THE OPERATING SYSTEM ZOO 35
 - 1.4.1 Mainframe Operating Systems 35
 - 1.4.2 Server Operating Systems 35
 - 1.4.3 Multiprocessor Operating Systems 36
 - 1.4.4 Personal Computer Operating Systems 36
 - 1.4.5 Handheld Computer Operating Systems 36
 - 1.4.6 Embedded Operating Systems 36
 - 1.4.7 Sensor-Node Operating Systems 37
 - 1.4.8 Real-Time Operating Systems 37
 - 1.4.9 Smart Card Operating Systems 38

- 1.5 OPERATING SYSTEM CONCEPTS 38
 - 1.5.1 Processes 39
 - 1.5.2 Address Spaces 41
 - 1.5.3 Files 41
 - 1.5.4 Input/Output 45
 - 1.5.5 Protection 45
 - 1.5.6 The Shell 45
 - 1.5.7 Ontogeny Recapitulates Phylogeny 46

- 1.6 SYSTEM CALLS 50
 - 1.6.1 System Calls for Process Management 53
 - 1.6.2 System Calls for File Management 56
 - 1.6.3 System Calls for Directory Management 57
 - 1.6.4 Miscellaneous System Calls 59
 - 1.6.5 The Windows Win32 API 60

- 1.7 OPERATING SYSTEM STRUCTURE 62
 - 1.7.1 Monolithic Systems 62
 - 1.7.2 Layered Systems 63
 - 1.7.3 Microkernels 65
 - 1.7.4 Client-Server Model 68
 - 1.7.5 Virtual Machines 68
 - 1.7.6 Exokernels 72

- 1.8 THE WORLD ACCORDING TO C 73
 - 1.8.1 The C Language 73
 - 1.8.2 Header Files 74
 - 1.8.3 Large Programming Projects 75
 - 1.8.4 The Model of Run Time 76

- 1.9 RESEARCH ON OPERATING SYSTEMS 77
- 1.10 OUTLINE OF THE REST OF THIS BOOK 78
- 1.11 METRIC UNITS 79
- 1.12 SUMMARY 80

2 PROCESSES AND THREADS

85

- 2.1 PROCESSES 85
 - 2.1.1 The Process Model 86
 - 2.1.2 Process Creation 88
 - 2.1.3 Process Termination 90
 - 2.1.4 Process Hierarchies 91
 - 2.1.5 Process States 92
 - 2.1.6 Implementation of Processes 94
 - 2.1.7 Modeling Multiprogramming 95
- 2.2 THREADS 97
 - 2.2.1 Thread Usage 97
 - 2.2.2 The Classical Thread Model 102
 - 2.2.3 POSIX Threads 106
 - 2.2.4 Implementing Threads in User Space 108
 - 2.2.5 Implementing Threads in the Kernel 111
 - 2.2.6 Hybrid Implementations 112
 - 2.2.7 Scheduler Activations 113
 - 2.2.8 Pop-Up Threads 114
 - 2.2.9 Making Single-Threaded Code Multithreaded 115
- 2.3 INTERPROCESS COMMUNICATION 119
 - 2.3.1 Race Conditions 119
 - 2.3.2 Critical Regions 121
 - 2.3.3 Mutual Exclusion with Busy Waiting 121
 - 2.3.4 Sleep and Wakeup 127
 - 2.3.5 Semaphores 130
 - 2.3.6 Mutexes 132

- 2.3.7 Monitors 137
- 2.3.8 Message Passing 144
- 2.3.9 Barriers 146
- 2.3.10 Avoiding Locks: Read-Copy-Update 148
- 2.4 SCHEDULING 148
 - 2.4.1 Introduction to Scheduling 149
 - 2.4.2 Scheduling in Batch Systems 156
 - 2.4.3 Scheduling in Interactive Systems 158
 - 2.4.4 Scheduling in Real-Time Systems 164
 - 2.4.5 Policy Versus Mechanism 165
 - 2.4.6 Thread Scheduling 165
- 2.5 CLASSICAL IPC PROBLEMS 167
 - 2.5.1 The Dining Philosophers Problem 167
 - 2.5.2 The Readers and Writers Problem 169
- 2.6 RESEARCH ON PROCESSES AND THREADS 172
- 2.7 SUMMARY 173

3 MEMORY MANAGEMENT 181

- 3.1 NO MEMORY ABSTRACTION 182
- 3.2 A MEMORY ABSTRACTION: ADDRESS SPACES 185
 - 3.2.1 The Notion of an Address Space 185
 - 3.2.2 Swapping 187
 - 3.2.3 Managing Free Memory 190
- 3.3 VIRTUAL MEMORY 194
 - 3.3.1 Paging 195
 - 3.3.2 Page Tables 198
 - 3.3.3 Speeding Up Paging 201
 - 3.3.4 Page Tables for Large Memories 205

- 3.4 PAGE REPLACEMENT ALGORITHMS 209
 - 3.4.1 The Optimal Page Replacement Algorithm 209
 - 3.4.2 The Not Recently Used Page Replacement Algorithm 210
 - 3.4.3 The First-In, First-Out (FIFO) Page Replacement Algorithm 211
 - 3.4.4 The Second-Chance Page Replacement Algorithm 211
 - 3.4.5 The Clock Page Replacement Algorithm 212
 - 3.4.6 The Least Recently Used (LRU) Page Replacement Algorithm 213
 - 3.4.7 Simulating LRU in Software 214
 - 3.4.8 The Working Set Page Replacement Algorithm 215
 - 3.4.9 The WSClock Page Replacement Algorithm 219
 - 3.4.10 Summary of Page Replacement Algorithms 221

- 3.5 DESIGN ISSUES FOR PAGING SYSTEMS 222
 - 3.5.1 Local versus Global Allocation Policies 222
 - 3.5.2 Load Control 225
 - 3.5.3 Page Size 225
 - 3.5.4 Separate Instruction and Data Spaces 227
 - 3.5.5 Shared Pages 228
 - 3.5.6 Shared Libraries 229
 - 3.5.7 Mapped Files 231
 - 3.5.8 Cleaning Policy 232
 - 3.5.9 Virtual Memory Interface 232

- 3.6 IMPLEMENTATION ISSUES 233
 - 3.6.1 Operating System Involvement with Paging 233
 - 3.6.2 Page Fault Handling 234
 - 3.6.3 Instruction Backup 235
 - 3.6.4 Locking Pages in Memory 236
 - 3.6.5 Backing Store 237
 - 3.6.6 Separation of Policy and Mechanism 239

- 3.7 SEGMENTATION 240
 - 3.7.1 Implementation of Pure Segmentation 243
 - 3.7.2 Segmentation with Paging: MULTICS 243
 - 3.7.3 Segmentation with Paging: The Intel x86 247

- 3.8 RESEARCH ON MEMORY MANAGEMENT 252

- 3.9 SUMMARY 253

4	FILE SYSTEMS	263
4.1	FILES	265
4.1.1	File Naming	265
4.1.2	File Structure	267
4.1.3	File Types	268
4.1.4	File Access	269
4.1.5	File Attributes	271
4.1.6	File Operations	271
4.1.7	An Example Program Using File-System Calls	273
4.2	DIRECTORIES	276
4.2.1	Single-Level Directory Systems	276
4.2.2	Hierarchical Directory Systems	276
4.2.3	Path Names	277
4.2.4	Directory Operations	280
4.3	FILE-SYSTEM IMPLEMENTATION	281
4.3.1	File-System Layout	281
4.3.2	Implementing Files	282
4.3.3	Implementing Directories	287
4.3.4	Shared Files	290
4.3.5	Log-Structured File Systems	293
4.3.6	Journaling File Systems	294
4.3.7	Virtual File Systems	296
4.4	FILE-SYSTEM MANAGEMENT AND OPTIMIZATION	299
4.4.1	Disk-Space Management	299
4.4.2	File-System Backups	306
4.4.3	File-System Consistency	312
4.4.4	File-System Performance	314
4.4.5	Defragmenting Disks	319
4.5	EXAMPLE FILE SYSTEMS	320
4.5.1	The MS-DOS File System	320
4.5.2	The UNIX V7 File System	323
4.5.3	CD-ROM File Systems	325
4.6	RESEARCH ON FILE SYSTEMS	331
4.7	SUMMARY	332

5 INPUT/OUTPUT 337

- 5.1 PRINCIPLES OF I/O HARDWARE 337
 - 5.1.1 I/O Devices 338
 - 5.1.2 Device Controllers 339
 - 5.1.3 Memory-Mapped I/O 340
 - 5.1.4 Direct Memory Access 344
 - 5.1.5 Interrupts Revisited 347

- 5.2 PRINCIPLES OF I/O SOFTWARE 351
 - 5.2.1 Goals of the I/O Software 351
 - 5.2.2 Programmed I/O 352
 - 5.2.3 Interrupt-Driven I/O 354
 - 5.2.4 I/O Using DMA 355

- 5.3 I/O SOFTWARE LAYERS 356
 - 5.3.1 Interrupt Handlers 356
 - 5.3.2 Device Drivers 357
 - 5.3.3 Device-Independent I/O Software 361
 - 5.3.4 User-Space I/O Software 367

- 5.4 DISKS 369
 - 5.4.1 Disk Hardware 369
 - 5.4.2 Disk Formatting 375
 - 5.4.3 Disk Arm Scheduling Algorithms 379
 - 5.4.4 Error Handling 382
 - 5.4.5 Stable Storage 385

- 5.5 CLOCKS 388
 - 5.5.1 Clock Hardware 388
 - 5.5.2 Clock Software 389
 - 5.5.3 Soft Timers 392

- 5.6 USER INTERFACES: KEYBOARD, MOUSE, MONITOR 394
 - 5.6.1 Input Software 394
 - 5.6.2 Output Software 399

- 5.7 THIN CLIENTS 416

- 5.8 POWER MANAGEMENT 417
 - 5.8.1 Hardware Issues 418

- 5.8.2 Operating System Issues 419
- 5.8.3 Application Program Issues 425

5.9 RESEARCH ON INPUT/OUTPUT 426

5.10 SUMMARY 428

6 DEADLOCKS 435

- 6.1 RESOURCES 436
 - 6.1.1 Preemptable and Nonpreemptable Resources 436
 - 6.1.2 Resource Acquisition 437
- 6.2 INTRODUCTION TO DEADLOCKS 438
 - 6.2.1 Conditions for Resource Deadlocks 439
 - 6.2.2 Deadlock Modeling 440
- 6.3 THE OSTRICH ALGORITHM 443
- 6.4 DEADLOCK DETECTION AND RECOVERY 443
 - 6.4.1 Deadlock Detection with One Resource of Each Type 444
 - 6.4.2 Deadlock Detection with Multiple Resources of Each Type 446
 - 6.4.3 Recovery from Deadlock 448
- 6.5 DEADLOCK AVOIDANCE 450
 - 6.5.1 Resource Trajectories 450
 - 6.5.2 Safe and Unsafe States 452
 - 6.5.3 The Banker's Algorithm for a Single Resource 453
 - 6.5.4 The Banker's Algorithm for Multiple Resources 454
- 6.6 DEADLOCK PREVENTION 456
 - 6.6.1 Attacking the Mutual-Exclusion Condition 456
 - 6.6.2 Attacking the Hold-and-Wait Condition 456
 - 6.6.3 Attacking the No-Preemption Condition 457
 - 6.6.4 Attacking the Circular Wait Condition 457
- 6.7 OTHER ISSUES 458
 - 6.7.1 Two-Phase Locking 458
 - 6.7.2 Communication Deadlocks 459

- 6.7.3 Livelock 461
- 6.7.4 Starvation 463
- 6.8 RESEARCH ON DEADLOCKS 464
- 6.9 SUMMARY 464

7 VIRTUALIZATION AND THE CLOUD 471

- 7.1 HISTORY 473
- 7.2 REQUIREMENTS FOR VIRTUALIZATION 474
- 7.3 TYPE 1 AND TYPE 2 HYPERVISORS 477
- 7.4 TECHNIQUES FOR EFFICIENT VIRTUALIZATION 478
 - 7.4.1 Virtualizing the Unvirtualizable 479
 - 7.4.2 The Cost of Virtualization 482
- 7.5 ARE HYPERVISORS MICROKERNELS DONE RIGHT? 483
- 7.6 MEMORY VIRTUALIZATION 486
- 7.7 I/O VIRTUALIZATION 490
- 7.8 VIRTUAL APPLIANCES 493
- 7.9 VIRTUAL MACHINES ON MULTICORE CPUS 494
- 7.10 LICENSING ISSUES 494
- 7.11 CLOUDS 495
 - 7.11.1 Clouds as a Service 496
 - 7.11.2 Virtual Machine Migration 496
 - 7.11.3 Checkpointing 497
- 7.12 CASE STUDY: VMWARE 498
 - 7.12.1 The Early History of VMware 498
 - 7.12.2 VMware Workstation 499

- 7.12.3 Challenges in Bringing Virtualization to the x86 500
 - 7.12.4 VMware Workstation: Solution Overview 502
 - 7.12.5 The Evolution of VMware Workstation 511
 - 7.12.6 ESX Server: VMware's type 1 Hypervisor 512
- 7.13 RESEARCH ON VIRTUALIZATION AND THE CLOUD 514

8 MULTIPLE PROCESSOR SYSTEMS 517

- 8.1 MULTIPROCESSORS 520
 - 8.1.1 Multiprocessor Hardware 520
 - 8.1.2 Multiprocessor Operating System Types 530
 - 8.1.3 Multiprocessor Synchronization 534
 - 8.1.4 Multiprocessor Scheduling 539
- 8.2 MULTICOMPUTERS 544
 - 8.2.1 Multicomputer Hardware 545
 - 8.2.2 Low-Level Communication Software 550
 - 8.2.3 User-Level Communication Software 552
 - 8.2.4 Remote Procedure Call 556
 - 8.2.5 Distributed Shared Memory 558
 - 8.2.6 Multicomputer Scheduling 563
 - 8.2.7 Load Balancing 563
- 8.3 DISTRIBUTED SYSTEMS 566
 - 8.3.1 Network Hardware 568
 - 8.3.2 Network Services and Protocols 571
 - 8.3.3 Document-Based Middleware 576
 - 8.3.4 File-System-Based Middleware 577
 - 8.3.5 Object-Based Middleware 582
 - 8.3.6 Coordination-Based Middleware 584
- 8.4 RESEARCH ON MULTIPLE PROCESSOR SYSTEMS 587
- 8.5 SUMMARY 588

9	SECURITY	593
9.1	THE SECURITY ENVIRONMENT	595
9.1.1	Threats	596
9.1.2	Attackers	598
9.2	OPERATING SYSTEMS SECURITY	599
9.2.1	Can We Build Secure Systems?	600
9.2.2	Trusted Computing Base	601
9.3	CONTROLLING ACCESS TO RESOURCES	602
9.3.1	Protection Domains	602
9.3.2	Access Control Lists	605
9.3.3	Capabilities	608
9.4	FORMAL MODELS OF SECURE SYSTEMS	611
9.4.1	Multilevel Security	612
9.4.2	Covert Channels	615
9.5	BASICS OF CRYPTOGRAPHY	619
9.5.1	Secret-Key Cryptography	620
9.5.2	Public-Key Cryptography	621
9.5.3	One-Way Functions	622
9.5.4	Digital Signatures	622
9.5.5	Trusted Platform Modules	624
9.6	AUTHENTICATION	626
9.6.1	Authentication Using a Physical Object	633
9.6.2	Authentication Using Biometrics	636
9.7	EXPLOITING SOFTWARE	639
9.7.1	Buffer Overflow Attacks	640
9.7.2	Format String Attacks	649
9.7.3	Dangling Pointers	652
9.7.4	Null Pointer Dereference Attacks	653
9.7.5	Integer Overflow Attacks	654
9.7.6	Command Injection Attacks	655
9.7.7	Time of Check to Time of Use Attacks	656
9.8	INSIDER ATTACKS	657
9.8.1	Logic Bombs	657
9.8.2	Back Doors	658
9.8.3	Login Spoofing	659

- 9.9 MALWARE 660
 - 9.9.1 Trojan Horses 662
 - 9.9.2 Viruses 664
 - 9.9.3 Worms 674
 - 9.9.4 Spyware 676
 - 9.9.5 Rootkits 680
- 9.10 DEFENSES 684
 - 9.10.1 Firewalls 685
 - 9.10.2 Antivirus and Anti-Antivirus Techniques 687
 - 9.10.3 Code Signing 693
 - 9.10.4 Jailing 694
 - 9.10.5 Model-Based Intrusion Detection 695
 - 9.10.6 Encapsulating Mobile Code 697
 - 9.10.7 Java Security 701
- 9.11 RESEARCH ON SECURITY 703
- 9.12 SUMMARY 704

10 CASE STUDY 1: UNIX, LINUX, AND ANDROID 713

- 10.1 HISTORY OF UNIX AND LINUX 714
 - 10.1.1 UNICS 714
 - 10.1.2 PDP-11 UNIX 715
 - 10.1.3 Portable UNIX 716
 - 10.1.4 Berkeley UNIX 717
 - 10.1.5 Standard UNIX 718
 - 10.1.6 MINIX 719
 - 10.1.7 Linux 720
- 10.2 OVERVIEW OF LINUX 723
 - 10.2.1 Linux Goals 723
 - 10.2.2 Interfaces to Linux 724
 - 10.2.3 The Shell 725
 - 10.2.4 Linux Utility Programs 728
 - 10.2.5 Kernel Structure 730
- 10.3 PROCESSES IN LINUX 733
 - 10.3.1 Fundamental Concepts 733
 - 10.3.2 Process-Management System Calls in Linux 735

- 10.3.3 Implementation of Processes and Threads in Linux 739
- 10.3.4 Scheduling in Linux 746
- 10.3.5 Booting Linux 751
- 10.4 MEMORY MANAGEMENT IN LINUX 753
 - 10.4.1 Fundamental Concepts 753
 - 10.4.2 Memory Management System Calls in Linux 756
 - 10.4.3 Implementation of Memory Management in Linux 758
 - 10.4.4 Paging in Linux 764
- 10.5 INPUT/OUTPUT IN LINUX 767
 - 10.5.1 Fundamental Concepts 767
 - 10.5.2 Networking 769
 - 10.5.3 Input/Output System Calls in Linux 770
 - 10.5.4 Implementation of Input/Output in Linux 771
 - 10.5.5 Modules in Linux 774
- 10.6 THE LINUX FILE SYSTEM 775
 - 10.6.1 Fundamental Concepts 775
 - 10.6.2 File-System Calls in Linux 780
 - 10.6.3 Implementation of the Linux File System 783
 - 10.6.4 NFS: The Network File System 792
- 10.7 SECURITY IN LINUX 798
 - 10.7.1 Fundamental Concepts 798
 - 10.7.2 Security System Calls in Linux 800
 - 10.7.3 Implementation of Security in Linux 801
- 10.8 ANDROID 802
 - 10.8.1 Android and Google 803
 - 10.8.2 History of Android 803
 - 10.8.3 Design Goals 807
 - 10.8.4 Android Architecture 809
 - 10.8.5 Linux Extensions 810
 - 10.8.6 Dalvik 814
 - 10.8.7 Binder IPC 815
 - 10.8.8 Android Applications 824
 - 10.8.9 Intents 836
 - 10.8.10 Application Sandboxes 837
 - 10.8.11 Security 838
 - 10.8.12 Process Model 844
- 10.9 SUMMARY 848

11	CASE STUDY 2: WINDOWS 8	857
11.1	HISTORY OF WINDOWS THROUGH WINDOWS 8.1	857
11.1.1	1980s: MS-DOS	857
11.1.2	1990s: MS-DOS-based Windows	859
11.1.3	2000s: NT-based Windows	859
11.1.4	Windows Vista	862
11.1.5	2010s: Modern Windows	863
11.2	PROGRAMMING WINDOWS	864
11.2.1	The Native NT Application Programming Interface	867
11.2.2	The Win32 Application Programming Interface	871
11.2.3	The Windows Registry	875
11.3	SYSTEM STRUCTURE	877
11.3.1	Operating System Structure	877
11.3.2	Booting Windows	893
11.3.3	Implementation of the Object Manager	894
11.3.4	Subsystems, DLLs, and User-Mode Services	905
11.4	PROCESSES AND THREADS IN WINDOWS	908
11.4.1	Fundamental Concepts	908
11.4.2	Job, Process, Thread, and Fiber Management API Calls	914
11.4.3	Implementation of Processes and Threads	919
11.5	MEMORY MANAGEMENT	927
11.5.1	Fundamental Concepts	927
11.5.2	Memory-Management System Calls	931
11.5.3	Implementation of Memory Management	932
11.6	CACHING IN WINDOWS	942
11.7	INPUT/OUTPUT IN WINDOWS	943
11.7.1	Fundamental Concepts	944
11.7.2	Input/Output API Calls	945
11.7.3	Implementation of I/O	948
11.8	THE WINDOWS NT FILE SYSTEM	952
11.8.1	Fundamental Concepts	953
11.8.2	Implementation of the NT File System	954
11.9	WINDOWS POWER MANAGEMENT	964

- 11.10 SECURITY IN WINDOWS 8 966
 - 11.10.1 Fundamental Concepts 967
 - 11.10.2 Security API Calls 969
 - 11.10.3 Implementation of Security 970
 - 11.10.4 Security Mitigations 972
- 11.11 SUMMARY 975

12 OPERATING SYSTEM DESIGN 981

- 12.1 THE NATURE OF THE DESIGN PROBLEM 982
 - 12.1.1 Goals 982
 - 12.1.2 Why Is It Hard to Design an Operating System? 983
- 12.2 INTERFACE DESIGN 985
 - 12.2.1 Guiding Principles 985
 - 12.2.2 Paradigms 987
 - 12.2.3 The System-Call Interface 991
- 12.3 IMPLEMENTATION 993
 - 12.3.1 System Structure 993
 - 12.3.2 Mechanism vs. Policy 997
 - 12.3.3 Orthogonality 998
 - 12.3.4 Naming 999
 - 12.3.5 Binding Time 1001
 - 12.3.6 Static vs. Dynamic Structures 1001
 - 12.3.7 Top-Down vs. Bottom-Up Implementation 1003
 - 12.3.8 Synchronous vs. Asynchronous Communication 1004
 - 12.3.9 Useful Techniques 1005
- 12.4 PERFORMANCE 1010
 - 12.4.1 Why Are Operating Systems Slow? 1010
 - 12.4.2 What Should Be Optimized? 1011
 - 12.4.3 Space-Time Trade-offs 1012
 - 12.4.4 Caching 1015
 - 12.4.5 Hints 1016
 - 12.4.6 Exploiting Locality 1016
 - 12.4.7 Optimize the Common Case 1017

- 12.5 PROJECT MANAGEMENT 1018
 - 12.5.1 The Mythical Man Month 1018
 - 12.5.2 Team Structure 1019
 - 12.5.3 The Role of Experience 1021
 - 12.5.4 No Silver Bullet 1021
- 12.6 TRENDS IN OPERATING SYSTEM DESIGN 1022
 - 12.6.1 Virtualization and the Cloud 1023
 - 12.6.2 Manycore Chips 1023
 - 12.6.3 Large-Address-Space Operating Systems 1024
 - 12.6.4 Seamless Data Access 1025
 - 12.6.5 Battery-Powered Computers 1025
 - 12.6.6 Embedded Systems 1026
- 12.7 SUMMARY 1027

13 READING LIST AND BIBLIOGRAPHY 1031

- 13.1 SUGGESTIONS FOR FURTHER READING 1031
 - 13.1.1 Introduction 1031
 - 13.1.2 Processes and Threads 1032
 - 13.1.3 Memory Management 1033
 - 13.1.4 File Systems 1033
 - 13.1.5 Input/Output 1034
 - 13.1.6 Deadlocks 1035
 - 13.1.7 Virtualization and the Cloud 1035
 - 13.1.8 Multiple Processor Systems 1036
 - 13.1.9 Security 1037
 - 13.1.10 Case Study 1: UNIX, Linux, and Android 1039
 - 13.1.11 Case Study 2: Windows 8 1040
 - 13.1.12 Operating System Design 1040
- 13.2 ALPHABETICAL BIBLIOGRAPHY 1041

INDEX 1071