

Operation Systems, *Third Edition*

By Gary Nutt

<u>PREFACE</u>	3
<u>To the Student</u>	3
<u>Topic Order</u>	6
<u>To the Instructor</u>	8
<u>About the Laboratory Environment</u>	9
<u>Acknowledgements</u>	10
<u>Changes in the Second Edition</u>	11
<u>Changes in the Third Edition</u>	11
1. INTRODUCTION	27
1.1 Computers and Software	28
General System Software	29
Resource Abstraction	30
Example: An Abstraction of a Disk Drive	32
Resource Sharing	33
1.2 Operating System Strategies	37
Batch Systems	38
Example: Batch Files	40
Timesharing Systems	40
Example: The UNIX Timesharing System	42
Personal Computers and Workstations	43
Example: The Microsoft Windows OS Family	45
Embedded Systems	46
Example: VxWorks	47
Small, Communicating Computers	48
Example: Windows CE (Pocket PC)	49
Networks	50
The Genesis of Modern Operating Systems	51
1.3 Summary	51
1.4 Exercises	51
2 USING THE OPERATING SYSTEM	53
2.1 The Programmer's Abstract Machine	53
Sequential Computation	54
Multithreaded Computation	55

2.2	Resources	56
	Using Files	56
	Example: POSIX Files	56
	Example: Windows Files	58
	Using Other Resources	59
2.3	Processes and Threads	60
	Creating Processes and Threads	61
	Example: Using FORK(), JOIN(), and QUIT()	62
2.4	Writing Concurrent Programs	63
	Multiple Single-Threaded Processes: The UNIX Model	64
	Example: Executing Commands in UNIX	65
	Multiple Processes and Multiple Threads Per Process: The Windows Model	69
	Example: Launching Windows Processes	71
2.5	Objects	75
2.6	Summary	76
2.7	Exercises	76
	Lab Exercise: A Simple Shell	77
	Background	78
	Attaching the Problem	81
	Lab Exercise: A Multithreaded Application	83
	Background	83
	Attacking the Problem	86
3	OPERATING SYSTEM ORGANIZATION	89
3.1	Basic Functions	89
	Device Management	89
	Process, Thread and Resource Management	90
	Memory Management	91
	File Management	91
3.2	General Implementation Considerations	92
	Performance	92
	Exclusive Use of Resources	93
	Processor Modes	93
	Kernels	94
	Requesting Services from the Operating System	95
	Software Modularization	96
3.3	Contemporary OS Kernels	97
	UNIX Kernels	98
	Example: Linux	99
	The Windows NT Executive and Kernel	100
3.4	Summary	102

3.5	<u>Exercises</u>	102
	Lab Exercise: Observing OS Behavior	103
	<u>Background</u>	104
	<u>Attacking the Problem</u>	104
4	<u>COMPUTER ORGANIZATION</u>	105
4.1	<u>The von Neumann Architecture</u>	106
	<u>Evolving to the von Neumann Architecture</u>	106
	<u>The Basic Idea</u>	106
4.2	<u>The Central Processing Unit</u>	108
	<u>The Arithmetical-Logical Unit</u>	108
	<u>The Control Unit</u>	109
	<u>Implementing the Processor</u>	110
4.3	<u>The Primary (Executable) Memory</u>	111
4.4	<u>I/O Devices</u>	112
	<u>Device Controllers</u>	113
	<u>Direct Memory Access</u>	114
	<u>Memory-mapped I/O</u>	114
4.5	<u>Interrupts</u>	115
	<u>The Trap Instruction Revisited</u>	118
4.6	<u>Conventional Contemporary Computers</u>	118
	<u>Bootstrapping the Machine</u>	119
4.7	<u>Mobile Computers</u>	120
	<u>System-on-a-Chip Technology</u>	121
	<u>Power Management</u>	122
	<u>Example: The Itsy Mobile Computer</u>	122
4.8	<u>Multiprocessors and Parallel Computers</u>	123
	<u>Parallel Instruction Execution</u>	123
	<u>Array Processors</u>	124
	<u>Shared Memory Multiprocessors</u>	125
	<u>Distributed Memory Multiprocessors</u>	125
	<u>Network of Workstations</u>	125
4.9	<u>Summary</u>	125
4.10	<u>Exercises</u>	126
5	<u>DEVICE MANAGEMENT</u>	130
5.1	<u>The I/O System</u>	130
	<u>Device Manager Abstraction</u>	131
	<u>I/O-Processor Overlap within an Application</u>	131

5.2	I/O Strategies	133
	Direct I/O with Polling	134
	Interrupt-driven I/O	134
	Polling Versus Interrupt-Driven I/O Performance	135
5.3	Device Manager Design	136
	Device Independent Driver Framework	137
	Servicing Interrupts	137
	Example: Linux Device I/O	138
5.4	Buffering	140
5.5	Device Class Characteristics	142
	Communication Devices	142
	Example: Asynchronous Serial Devices	144
	Sequentially Accessed Storage Devices	144
	Example: Traditional Magnetic Tape	145
	Randomly Accessed Storage Devices	145
	Example: Magnetic Disk	146
	Performance: Optimizing Access on Magnetic Disks	147
	Example: CD-ROM and DVD	149
5.6	Summary	151
5.7	Exercises	151
	Lab Exercise: A Floppy Disk Driver	152
	Part A	152
	Part B	153
	Part C	153
	Background	153
	Attacking the Problem	156
6	IMPLEMENTING PROCESSES, THREADS, AND RESOURCES	159
6.1	The Task at Hand	159
	The Abstract Machine for Classic Processes	160
	Supporting Modern Processes and Threads	161
	Resources	162
	The Process Address Space	162
	OS Families	163
	Process Manager Responsibilities	163
6.2	The Hardware Process	164
6.3	The Abstract Machine Interface	165
6.4	The Process Abstraction	168
	Example: Linux Process Descriptor	169
	Example: Windows NT/2000/XP Process Descriptors	171
6.5	The Thread Abstraction	171

Example: Windows NT/2000/XP Thread Descriptors	172
6.6 State Diagrams	173
Example: UNIX State Diagram	174
6.7 Resource Managers	175
6.8 Generalizing Process Management Policies	177
Refining the Process Manager	177
Specializing Resource Allocation Strategies	178
6.9 Summary	178
6.10 Exercises	179
LAB EXERCISE 1: Kernel Timers	180
Background	181
Attacking the Problem	183
Lab Exercise 2: Manipulating Kernel Objects	187
Background	187
Attacking the Problem	194
7 SCHEDULING	196
7.1 Overview	196
7.2 Scheduling Mechanisms	197
The Process Scheduler Organization	197
Saving the Context	198
Voluntary CPU Sharing	198
Involuntary CPU Sharing	200
Performance	200
7.3 Strategy Selection	201
Scheduler Characteristics	201
A Model to Study Scheduling	202
Analysis: Partitioning a Process into Small Processes	204
7.4 Nonpreemptive Strategies	204
Analysis: Approximating System Load	204
First-Come-First-Served	205
Analysis: Predicting Wait Times for FCFS	206
Shortest Job Next	207
Priority Scheduling	208
Deadline Scheduling	209
7.5 Preemptive Strategies	209
Round Robin	210
Multiple-level Queues	212
7.6 Implementing the Scheduler	213
Example: The Linux Scheduling Mechanism	214

Example: BSD UNIX Scheduling Policy	215
Example: Windows NT/2000/XP Thread Scheduling	215
7.7 Summary	216
7.8 Exercises	216
Lab Exercise: Analyzing the Round Robin Scheduling	218
Background	219
Attacking the Problem	222
8 BASIC SYNCHRONIZATION PRINCIPLES	224
8.1 Cooperating Processes	224
Critical Sections	227
Deadlock	231
Resource Sharing	232
8.2 Evolving from the Classic Solution	233
8.3 Semaphores: The Basis of Modern Solutions	235
Principles of Operation	236
Example: Using Semaphores	237
Practical Considerations	242
8.4 Synchronization in Shared Memory Multiprocessors	246
8.5 Summary	246
8.6 Exercises	246
Lab Exercise: Bounded Buffer Problem	249
Background	250
Attacking the Problem	257
9 HIGH-LEVEL SYNCHRONIZATION AND INTERPROCESS COMMUNICATION	258
9.1 Alternative Synchronization Primitives	259
AND Synchronization	260
Example: Using AND Synchronization to Solve the Dining Philosophers Problem	261
Events	262
Example: Using Generic Events	262
Example: Windows NT/2000/XP Dispatcher Objects	263
9.2 Monitors	264
Principles of Operation	264
Condition Variables	265
Example: Using Monitors	268
Some Practical Aspects of Using Monitors	271
9.3 Interprocess Communication	271

The Pipe Model	272
Message Passing Mechanisms	272
Mailboxes	273
Message Protocols	273
Using the send() and receive() Operations	274
Example: Synchronized IPC	275
Deferred Message Copying	275
9.4 Summary	276
9.5 Exercises	276
Lab Exercise: Using Pipes	278
Background	279
Attacking the Problem	284
Lab Exercise: Refining the Shell	284
Background	285
Attacking the Problem	286
10 DEADLOCK	287
10.1 Background	288
Prevention	290
Avoidance	290
Detection and Recovery	291
Manual Deadlock Management	291
10.2 A System Deadlock Model	291
Example: Single Resource Type	293
10.3 Prevention	294
Hold and Wait	294
Circular Wait	295
Allowing Preemption	296
10.4 Avoidance	297
The Banker's Algorithm	298
Example: Using the Banker's Algorithm	299
10.5 Detection and Recovery	300
Serially Reusable Resources	301
Example: Serially Reusable Resource Graphs	304
Consumable Resources	305
General Resource Systems	307
Recovery	307
10.6 Summary	308
10.7 Exercises	308
11 MEMORY MANAGEMENT	311

11.1	The Basics	311
11.2	The Address Space Abstraction	313
	Managing the Address Space	314
	Example: Static Address Binding	315
	Dynamic Memory for Data Structures	319
	Modern Memory Binding	319
11.3	Memory Allocation	320
	Fixed-partition Memory Strategies	321
	Variable-partition Memory Strategies	321
	Analysis: The Cost of Moving Programs	323
	Contemporary Allocation Strategies	324
11.4	Dynamic Address Space Binding	325
	Runtime Bound Checking: The Isolation Mechanism	327
11.5	Modern Memory Manager Strategies	327
	Swapping	328
	Virtual Memory	329
	Example: Using Cache Memory	330
	Shared-memory Multiprocessors	331
11.6	Summary	333
11.7	Exercises	333
	Lab Exercise: Using Shared Memory	335
	Background	336
	Attacking the Problem	339
12	VIRTUAL MEMORY	341
12.1	The Task at Hand	341
12.2	Address Translation	342
	Address Space Mapping	342
	Segmentation and Paging	343
12.3	Paging	344
	Paging Virtual Address Translation	345
	Example: Contemporary Page Table Implementations	348
12.4	Static Paging Algorithms	348
	The Fetch Policy	350
	Demand Paging Algorithms	350
	Stack Algorithms	353
	Implementing LRU	354
	Paging Performance	356
12.5	Dynamic Paging Algorithms	357
	The Working Set Algorithm	357

Implementing the Working Set Algorithm	360
Performance Consideration: Taking Advantage of Paging with IPC	362
Example: Windows NT/2000 Virtual Memory	362
Example: Linux Virtual Memory	364
12.6 Segmentation	365
Address Translation	366
Implementation	367
Example: The Multics Segmentation System	369
12.7 Memory-mapped Files	370
12.8 Summary	372
12.9 Exercises	372
Lab Exercise: Memory-mapped Files	374
Background	375
Attacking the Problem	377
13 FILE MANAGEMENT	379
13.1 The Task at Hand	380
13.2 Files	381
Low-level Files	383
Structured Files	384
Database Management Systems	388
Multimedia Storage	388
13.3 Low-level File Implementations	388
The open() and close() Operations	389
Example: UNIX open and close	391
Block Management	392
Example: UNIX File Structure	394
Example: The DOS FAT File System	395
Reading and Writing the Byte Stream	397
13.4 Supporting High-level File Abstractions	399
Structured Sequential Files	399
Indexed Sequential Files	400
Database Management Systems	400
Multimedia Documents	400
13.5 Directories	401
Directory Structures	401
Example: Some Directory Approaches	402
13.6 Implementing Directories	404
Directory Entries	404
Opening a File	405
13.7 File Systems	405

Example: The ISO 9660 File System	405
Mounting File Systems	407
Heterogeneous File Systems	408
13.8 Summary	409
13.9 Exercises	409
Lab Exercise: A Simple File Manager	412
Background	413
Attacking the Problem	415
14 PROTECTION AND SECURITY	418
14.1 The Problem	419
The Goal	420
Policy and Mechanism	420
Context for Protection and Security	421
The Cost of Protection Mechanisms	423
14.2 Authentication	423
External User Authentication	423
Example: Windows NT/2000/XP User Authentication	426
Internal Thread/Process Authentication	428
Authentication in the Network	429
Software Authentication	431
14.3 Authorization	432
Ad Hoc Authorization Mechanisms	433
A General Model for Authorization	435
Implementing Security Policies	437
Implementing General Authorization Mechanisms	438
Protection Domains	438
Implementing the Access Matrix	440
14.4 Cryptography	442
The Big Picture	443
Private Key Encryption	443
Public Key Encryption	445
Example: PGP Encryption	446
Internet Content Delivery	446
14.5 Summary	447
14.6 Exercises	447
15 NETWORKS	449
15.1 From Computer Communications to Networks	449
Switched Networks	450
Network Hardware Requirements	451
Network Software Requirements	451

15.2	The ISO OSI Network Architecture Model	453
	The Evolution of Network Protocols	453
	The ISO OSI Model	454
15.3	Media Access Control (MAC) Protocols	456
	The Physical Layer	456
	SIDEBAR: Fast Physical Layers	457
	The Data Link Layer	458
	Contemporary Networks	459
15.4	The Network Layer	461
	Internet Addresses	462
	Routing	463
	Using the Network Layer	464
	Sidebar box: Latency in the Internet	464
15.5	The Transport Layer	465
	Communication Ports	465
	Data Types	466
	Reliable Communication	467
	Performance Considerations: Datagrams and Virtual Circuits	467
15.6	Using the Transport Layer	468
	Naming and Addresses	468
	Example: The Domain Name Service	469
	The Client-server Model	470
15.7	Network Security	472
	Transport Layer Security: Firewalls	472
	Network Layer Security: IPsec	473
15.8	Summary	473
15.9	Exercises	474
	Lab Exercise: Using TCP/IP	475
	Background	475
	Attacking the Problem	480
16	REMOTE FILES	481
16.1	Sharing Information Across the Network	482
	Explicit File Copying Systems	483
	A Seamless File System Interface	484
	Distributing the Work	485
16.2	Remote Disk Systems	486
	Remote Disk Operation	487
	Performance Considerations	488
	Reliability	489
	The Future of Remote Disks	491
16.3	Remote File Systems	492

The General Architecture	492
Block Caching	493
Crash Recovery	495
16.4 File-level Caching	497
The Andrew File System	497
The LOCUS File System	498
16.5 Directory Systems and Their Implementations	499
Filenames	499
Opening a File	501
16.6 Summary	502
16.7 Exercises	502
17 DISTRIBUTED COMPUTING	504
17.1 Distributed OS Mechanisms	505
17.2 Distributed Primary Memory	507
Remote Memory	509
Example: The Linda Programming Language	509
Distributed Shared Memory	511
17.3 Remote Procedure Call	512
How Does RPC Work?	512
Implementing RPC	513
17.4 Remote Objects	516
17.5 Distributing Process Management	518
General Process Management	519
Process and Thread Creation	519
Scheduling	519
Migration and Load Balancing	520
Distributed Synchronization	520
17.6 Summary	525
17.7 Exercises	525
Lab Exercise: Using Remote Procedure Call	526
Background	527
Attacking the Problem	532
18 DISTRIBUTED PROGRAMMING RUNTIME SYSTEMS	533
18.1 Supporting Distributed Software with Middleware	533
18.2 Classic Distributed Application Programs	534

18.3	Middleware Support for Classic Distributed Programming	535
	PVM	536
	The Beowulf Cluster Computing Environment	538
	The OSF Distributed Computing Environment	539
18.4	Distributed Programming on the Web	544
18.5	Middleware Support for Mobile Code	545
	Java and the Java Virtual Machine	546
	The ECMA-335 Common Language Infrastructure	550
18.6	Summary	554
18.7	Exercises	555
19	DESIGN STRATEGIES	556
19.1	Design Considerations	557
	Performance, Performance, Performance	557
	Trusted Software	558
	Modularization	558
	Portability	560
19.2	Monolithic Kernels	560
	Example: MS-DOS	561
	Example: The UNIX Kernel	561
19.3	Modular Organization	562
	Example: Choices: An Object-oriented OS	562
19.4	Extensible Nucleus, or Microkernel, Organization	564
	Example: The Mach Operating System	565
19.5	Layered Organizations	570
19.6	Operating Systems for Distributed Systems	571
	Network Operating Systems	571
	Example: BSD UNIX	571
	Distributed Operating Systems	572
	Example: The CHORUS Operating System	573
19.7	Summary	575
19.8	Exercises	576
20	THE LINUX KERNEL	577
20.1	The Linux Kernel	577
20.2	Kernel Organization	577
	Using Kernel Services	578

Starting the Kernel	579
Control Flow in the Machine	580
20.3 Modules and Device Management	580
Module Organization	580
Module Installation and Removal	581
20.4 Process and Resource Management	581
Running the Process Manager	582
Creating a New Task	583
IPC and Synchronization	584
The Scheduler	584
20.5 Memory Manager	585
The Virtual Address Space	587
The Page Fault Handler	587
20.6 File Management	588
20.7 Summary	590
21 THE WINDOWS NT/2000XP KERNEL	591
21.1 Introduction	591
21.2 The NT Kernel	592
Objects	592
Threads	593
Multiprocess Synchronization	593
Traps, Interrupts, and Exceptions	593
21.3 The NT Executive	594
Object Manager	594
Process and Thread Manager	595
Virtual Memory Manager	596
I/O Manager	596
The Cache Manager	598
21.4 Kernel Local Procedure Calls and IPC	599
The Native API	600
21.5 Subsystems	600
21.6 Summary	601
GLOSSARY	602
BIBLIOGRAPHY	628
Index 634	