
Index

- /*, 38–39, 348
 - ...*, 125, 307, 315, 327, 339–340, 345, 621
 - ..*, 307, 315, 339–340, 621
 - #!*, 62
- A**
- ABI. See *application binary interface*
 - absolute pathname, 39, 615, 635
 - accept* system call, 440, 453, 455–457, 469, 540, 564
 - definition, 439
 - access control, filesystem, 41
 - access control list, 41, 125, 301, 615
 - access rights, 450, 460
 - receiving, 463
 - access* system call, 254
 - access* vnode operator, 296
 - accounting, process resource, 58–59, 73–74, 111
 - accton**, 608
 - ACL. See *access control list*
 - ACPI. See *advanced configuration and power interface*
 - active page list, 188
 - ad_done()*, 281
 - address family, 450–451, 454, 473, 615
 - address resolution protocol, 492, 509–510, 567, 615–616
 - implementation of, 509–510
 - purpose of, 509
 - address, socket, 453–454
 - address space. See *virtual address space*
 - address structure
 - Internet, 454
 - local domain, 454
 - socket, 438, 645
 - address translation, 136, 615
 - addresses, IPv6, 560–561
 - adjtime* system call, 66
 - administrative control, 123–129
 - ad_start()*, 280
 - adstrategy()*, 280–281
 - advanced configuration and power interface, 266, 289
 - advanced programmable interrupt controller, 265, 268, 600
 - advisory locking, 232, 296, 615
 - advlock* vnode operator, 296
 - AF_INET, 438, 564
 - AF_INET6, 564
 - AH. See *authentication header*
 - ahc_action()*, 279
 - ahc_done()*, 279
 - aio_error* system call, 230–231
 - aio_read* system call, 230, 260
 - aio_return* system call, 231
 - aio_suspend* system call, 231

- aio_waitcomplete* system call, 231
 - aio_write* system call, 230, 260
 - algorithm
 - for *disksort*(), 224
 - elevator sorting, 224
 - mbuf storage-management, 447–448
 - for physical I/O, 221
 - TCP, 534–541
 - TCP slow-start, 550–554
 - Allman, Eric, xxvii
 - allocation
 - descriptor, 457
 - directory space, 308–309
 - FFS file block, 369, 372–375, 616
 - FFS fragment, 374–375
 - inode, 299
 - kernel address space, 146–150
 - kernel memory, 32
 - kernel resource, 165–166
 - mbuf, 447–448
 - process identifier, 109
 - virtual memory map, 204–206
 - allocator, zone, 149–150
 - allocbuf*(), 251–252
 - alldirect* structure, 331–332, 334–335, 337
 - allocindir* structure, 332, 336–338
 - ancillary data, 440, 458, 460, 615
 - Andrew Filesystem, 386
 - anonymous object, 151, 154, 616
 - ANSI, 266
 - AOUT executable format, 61
 - API. See *application programming interface*
 - APIC. See *advanced programmable interrupt controller*
 - append-only file, 304
 - Apple OS/X operating system, 3, 300, 309
 - application binary interface, 281–282
 - application, client-server, 42
 - application programming interface, 44, 281
 - architecture, PC, 264–266
 - arguments, marshalling of, 388, 632
 - ARP. See *address resolution protocol*
 - ARPANET, 6, 514
 - arpintr*(), 510
 - arpresolve*(), 510
 - assembly language in the kernel, 25, 54, 104, 218, 595–598, 600–601
 - AST. See *asynchronous system trap*
 - asymmetric cryptography, 581, 616
 - asynchronous I/O, 229–231
 - asynchronous system trap, 105, 616
 - ATA disk, 24, 265–267, 279–281, 286–289
 - I/O request, 280–281
 - layer, 279–281
 - ATA. See *ATA disk*
 - ata_completed*(), 281
 - ata_finish*(), 281
 - ata_interrupt*(), 281
 - ATAPI devices, 267–268
 - ATAPI. See *ATAPI devices*
 - ata_start*(), 280
 - ata_transaction*(), 280
 - AT&T, 7, 9, 11–12
 - ATTACHED flag, 331
 - definition, 331
 - attribute manipulation, filesystem, 296
 - attribute update, filestore, 358
 - authentication data, 573
 - authentication header, 563, 571–574, 578
 - autoconfiguration, 281, 559, 565, 602, 616
 - 4.4BSD, 281
 - contribution of, 8
 - data structures, 286–289
 - device driver support for, 217, 282–292
 - IPv6, 565–569
 - phase, 283
 - resource, 289–292
- ## B
- B programming language, 4
 - Babaoğlu, Özalp, 6
 - Bach, Maurice, xxi
 - background **fsck**, 356–357
 - background process, 122, 420, 616, 624
 - backing storage, 135, 616
 - basic input-output system, 225, 266, 595–596
 - PNP, 266
 - basic services, 598–600
 - bawrite*(), 249
 - bcopy*(), 208
 - BCPL, 4
 - bdwrite*(), 249
 - Bell Laboratories, 3–5
 - benefit of global vnode table, 246
 - Berkeley Software Design Inc., 11–13

- bind* system call, 519, 564
 - definition, 439
 - biod**, 396
 - biodone()*, 279, 281, 334
 - BIOS. See *basic input-output system*
 - bitmap dependencies, soft updates, 332
 - blkatoff* vnode operator, 359
 - block clustering, 360, 368–369, 375–379, 618
 - block I/O, 360–362
 - block size, 216, 363, 616
 - bmsafemap* structure, 332–334, 336, 338
 - Bolt Beranek and Newman, 45
 - boot**, 593–597, 610, 614, 624
 - /boot/device.hints**, 283
 - flags, 595
 - /boot/kernel/kernel**, 593–594
 - operation of, 595–596
 - bootstrapping, 25, 46, 593, 595–596, 617
 - setting time when, 65
 - see also **boot**
 - boot_time*, 408
 - bottom half of, 51, 617
 - device driver, 217
 - kernel, 51–52
 - terminal driver, 415
 - bqrelse()*, 249
 - bread()*, 223, 249, 251–252
 - break character, 426
 - breakpoint fault, 130, 617
 - brlse()*, 249
 - bremfree()*, 251
 - broadcast message, 481, 521, 526, 617
 - address, 481, 516–517
 - IP handling of, 523
 - broadcast storms, 526
 - BSD, 272
 - obtaining, xxiii
 - open source, 9–13
 - BSDI. See *Berkeley Software Design Inc.*
 - bss segment, 61, 617
 - b_to_q()*, 422, 425
 - buf* structure, 223
 - bufdaemon*, 50, 606
 - buffer cache, 299, 360–362, 617
 - 4.4BSD, 202
 - consistency, 252
 - effectiveness, 248
 - implementation of, 251–252
 - interface, 249
 - management, 248–252
 - memory allocation, 252
 - structure of, 250–251
 - buffer list
 - CLEAN, 251, 260
 - DIRTY, 250–251, 260
 - EMPTY, 251
 - LOCKED, 250
 - buffer update*, 334
 - buffer wait*, 334–335, 338, 341–342, 344
 - buffering
 - filesystem, 360–362
 - network, 505–506
 - policy, protocol, 505
 - terminal, 421–422
 - bus_add_child()*, 291–292
 - bus_child_detached()*, 291–292
 - bus_driver_added()*, 291
 - bus_probe_nomatch()*, 292
 - bus_read_ivar()*, 292
 - bus_write_ivar()*, 292
 - bwrite()*, 223, 249
 - bzero()*, 208
- ## C
- C-block, 421–422, 618
 - C library, 65
 - system calls in the, 54
 - C-list, 421–422, 425, 427, 430–431, 618
 - C programming language, 3–4, 26, 54
 - cache
 - buffer, 299, 360–362, 617
 - directory offset, 310–312
 - filename, 247–248
 - inode, 306–307
 - page list, 189
 - vnode, 155
 - callback, 83, 395, 403, 452, 617
 - callout queue, 59–61, 535, 602, 617
 - CAM. See *common access method*
 - camisr()*, 279
 - canonical mode, 43, 414, 617
 - canrecurse* flag, 98
 - castor oil, 421
 - catq()*, 422
 - caught signal, 28, 113, 618

- CCB. See *common access method control block*
- CD-ROM, xxiii, 11, 37, 258–259, 268, 277, 280
- CD9660 filesystem, 259
- CDB. See *command descriptor block*
- cdevsw* structure, 216, 219, 221–222
- character device, 219–223, 618
 - driver, 219
 - ioctl, 222
 - operations, 222
- character device interface, 215–216, 219, 221–223, 415, 618
- character-oriented device, 221–223
- character processing, 43
- chdir* system call, 39, 620
- checksum, 515, 519–521, 524, 542, 547, 618
- chflags* system call, 303, 350
- child process, 26, 85, 108, 618
- chkdq()*, 317–319, 369
- chmod* system call, 40, 350
- Chorus operating system, 22
- chown* system call, 40, 350
- chroot* system call, 39, 124–127, 640
- CIDR. See *classless inter-domain routing*
- CIFS. See *common Internet filesystem*
- classless inter-domain routing, 515–516, 558, 561, 633
- CLEAN buffer list, 251, 260
- client process, 42, 618
- client server
 - application, 42
 - interaction, NFS, 397–398
 - model, 455
 - programming, 438
- clock
 - alternate, 59
 - interrupt handling, 57–59
 - interrupt rate, 58
 - real-time, 50
- clock_skew*, 404, 408
- cloning route, 498, 509, 618
- close-on-exec, 228–230
- close* system call, 33, 228, 231, 247, 253, 402–403, 416, 441, 463–464, 541
- close* vnode operator, 296
- closedir()*, 309
- clustering, block, 360, 368–369, 375–379, 618
 - clustering, page, 174, 184, 191, 209, 618
 - cold start, 593, 618
 - coloring, page, 186–187
 - command descriptor block, 277
 - common access method, 24, 266–268, 271, 276–279, 283, 292
 - control block, 277–279
 - I/O request, 277–279
 - layer, 276–279
 - common Internet filesystem, 386–387
 - communication domain, 44, 436, 449–450, 619
 - data structures, 450
 - communication protocol. See *protocol*
 - COMPLETE flag, 331, 334, 336, 338
 - definition, 331
 - Computer Systems Research Group, xx, xxvii, 3, 7–10, 12–16, 45
 - config**, 282, 286, 619
 - configuration
 - file, 609, 619
 - kernel, 609–610
 - congestion control
 - network, 505–506
 - TCP, 550–554
 - see also *network buffering*
 - congestion window, 399, 552, 556
 - connect request, 485, 619
 - connect* system call, 439, 455, 457, 519–520, 522, 538, 564, 579, 619
 - definition, 439
 - connection
 - queueing, socket, 452, 455
 - setup, TCP, 529–530, 538–541
 - shutdown, TCP, 530, 541
 - states, TCP, 529–531
 - contents update, filestore, 358
 - context switching, 55, 80, 88–99, 619
 - involuntary, 89, 104
 - thread state, 89
 - voluntary, 89–93
 - continuation style, 579, 619
 - control-output routine, protocol, 488
 - control request, 487, 619
 - controlling process, 70, 120, 619
 - controlling terminal, 29, 70, 120, 619
 - revocation of, 247
 - cooked mode, 414
 - copy object, 4.4BSD, 163

- copy-on-write, 6, 31, 166, 209, 619
 - core file, 28, 113, 620
 - coredump()*, 117
 - cpu_exit()*, 111
 - cpu_switch()*, 104
 - crash dump, 88, 217, 223, 610–612, 620, 635
 - crash recovery, NFS, 408–409
 - crash, system, 40, 42, 223, 231, 233, 249, 284, 319, 324, 326, 328, 347, 357, 362–363, 380, 391–392, 396–399, 402–404, 408–410, 529, 531, 536, 599, 603, 608, 611–612, 620
 - create* vnode operator, 295, 297
 - creation and deletion, filestore, 358
 - credential, 66–69, 84, 109, 131, 164, 256, 391, 400–401, 620
 - cron**, 608
 - crypto_done()*, 583
 - crypto_freesession()*, 583
 - cryptographic descriptor, 582
 - cryptology
 - asymmetric, 581, 616
 - session, 581
 - subsystem, 581–583
 - symmetric, 581, 647
 - crypto_invoke()*, 583
 - crypto_newsession()*, 582–583
 - crypto_proc()*, 583
 - crypto_register()*, 582
 - csh** shell, 122
 - CSRG. See *Computer Systems Research Group*
 - CTSS operating system, 4
 - current working directory, 39, 314, 620
 - cur sig()*, 115, 117
 - cv_broadcast()*, 99
 - cv_signal()*, 99
 - cv_timedwait()*, 99
 - cv_timedwait_sig()*, 99
 - cv_wait()*, 99
 - cv_waitq_remove()*, 99
 - cv_wait_sig()*, 99
- D**
- dadone()*, 279
 - daemon, 620
 - NFS, 394–397
 - process, 233, 620
 - routing, 503
 - T-shirt, xxiv
 - DARPA. See *Defense Advanced Research Projects Agency*
 - dastart()*, 278
 - dastrategy()*, 278–279
 - data-carrier detect, 422–423
 - data-communications equipment, 422, 648
 - data segment, 29, 61, 63, 169, 620
 - expansion, 169
 - data structures
 - autoconfiguration, 286–289
 - communication domain, 450
 - interprocess communication, 449–455
 - socket, 451–453
 - data-terminal equipment, 422–423, 648
 - data-terminal ready, 422–423, 429
 - datagram socket, 437, 620
 - DCD. See *data-carrier detect*
 - DCE. See *data-communications equipment*
 - dead filesystem, 247
 - deadlock, 201
 - avoidance during *fork* system call, 166
 - avoidance when locking resources, 94–96, 465–466, 470
 - detection, 320–321, 381
 - memory, 165, 178–180, 194–195
 - network, 233–234
 - snapshot, 353–354
 - debugging
 - gdb**, 129, 611
 - information in exec header, 63
 - process, 116, 129–131
 - system, 611–612
 - see also *ptrace* system call
 - decapsulation, 474, 479, 620
 - decision, local-remote, 496
 - default pager, 177
 - Defense Advanced Research Projects Agency, 6, 8, 45, 514, 620
 - steering committee, 7
 - definition
 - ATTACHED flag, 331
 - COMPLETE flag, 331
 - DEPCOMPLETE flag, 331
 - defrtrlist_update()*, 566
 - delayed write, 249, 325, 348
 - demand paging. See *paging*

- denial of service attack, 540, 621
- DEPCOMPLETE flag, 331–332, 334, 336, 338, 340
 - definition, 331
- dependencies, soft updates, 325–329
- dependencies, virtual memory machine, 196–209
- descriptor, 33, 621
 - allocation, 457
 - duplication, 229–230
 - management of, 34–35, 226–230
 - multiplexing, 233–236
 - table, 34, 226, 621
 - use of, 33–34
- design
 - 4.2BSD IPC, 8
 - FreeBSD IPC, 436, 441
 - I/O system, 33–37
 - mbuf, 446–447
 - memory-management, 30–32
 - network, 45
 - NFS, 387–388
- /dev**, 35, 216, 268–269, 286, 292
 - /dev/console**, 607
 - /dev/fd**, 259
 - /dev/kmem**, 304, 612–613
 - /dev/mem**, 219, 223, 304
 - /dev/null**, 219
 - operation of, 268–269
- devclass*, 286
- devd**, 292
- development model, FreeBSD, 14–17
- device, 35, 37–38, 286
 - character-oriented, 221–223
 - configuration, 281–292
 - filesystem, 269, 271
 - flags, 423, 621
 - identification, 284–286
 - interrupt, 56–57, 628
 - interrupt handler, 56
 - module initialization, 601–603
 - number, 216, 621
 - overview, 263–269
 - pager, 154, 175–176
 - probing, 217
 - raw, 220–221
 - special file, 35, 621
 - swap, 140, 647
- device driver, 35, 215–217, 621, 644
 - attach routine, 284–286
 - bottom half of, 217
 - code for *select* system call, 238
 - interrupt handling, 218
 - maximum transfer size, 219
 - probe routine, 285
 - sections of a, 217
 - support for autoconfiguration, 217, 282–292
 - support for *select* system call, 222, 236–237
 - top half of, 217
- device_attach()*, 285–286
- device_identify()*, 285
- device_probe()*, 285
- device_t*, 269
- devinfo**, 289, 293
- dev_t*, 269
- df**, 346
- diradd* structure, 334, 338–340, 343
- DIRCHG flag, 343
- direct block dependencies, soft updates, 334–335
- direct memory access, 221, 276, 279–281, 292, 621–622
- direct route, 496
- directory, 38, 307, 621
 - dependencies, soft updates, 338–341
 - entry, 38, 298, 621
 - offset cache, 310–312
 - operations, 40–41
 - space allocation, 308–309
 - structure, 308–312
 - table, 198, 622
- dirrem* structure, 331, 341, 343–344
- DIRTY buffer list, 250–251, 260
- disk
 - device, 223–225
 - device interface, 223
 - device operations, 223
 - label, 225
 - partition, 217, 225, 359, 622, 632
 - SCSI, 277
 - structure, FFS, 363–366
 - subsystem, 266–268
- disksort()*, 224–225, 278, 280
 - algorithm for, 224

distributed filesystem, 40
 distributed program, 435
 DMA. See *direct memory access*
 DNS. See *domain name system*
doadump(), 611
 domain name system, 564
 domain. See *communication domain*
 double indirect block, 299, 622, 627
dquot structure, 317–319
 DTE. See *data-terminal equipment*
dtom(), 446, 448–449
 DTR. See *data-terminal ready*
dump, 220, 303, 358
 live, 358
dumpsys(), 611
dup system call, 35, 41, 228–229, 621, 624
 implementation of, 229
dup2 system call, 35, 229, 624
 duplication, process virtual memory,
 166–167

E

EAGAIN system error, 110, 229, 427, 460,
 465–466, 634
 effective GID. See *effective group identifier*
 effective group identifier, 67, 622
 effective UID. See *effective user identifier*
 effective user identifier, 67, 622
 Eighth Edition UNIX, 5
 EINTR system error, 55, 87, 110
 elevator sorting algorithm, 224, 622
 ELF executable format, 61
 Elz, Robert, 9, 316
 EMPTY buffer list, 251
 encapsulating-security payload, 563,
 571–572, 574, 578
 encapsulating security protocol, IPSec, 574
 encapsulation, 474, 479, 622
 entry to kernel, 52–53
 environment, location of process, 64
 epoch, 65
 erase character, 414, 623
 errno, 26, 54, 429, 576, 623
 ESP. See *encapsulating-security payload*
/etc, 348
 /etc/defaults/rc.conf, 607
 /etc/exports, 394

/etc/gettytab, 609
/etc/group, 609
/etc/master.passwd, 609
/etc/rc, 304, 607
/etc/rc.conf, 607
/etc/rc.d, 607–608
/etc/ttys, 607
ether_demux(), 510
 Ethernet, 6, 45, 476, 514
 event handler, 600–604, 623
 eviction notice, 406, 623
 exec header, 61
exec system call, 27, 34, 64, 67–69, 73, 79,
 108, 119–120, 130–131, 146, 164,
 166–168, 172, 174, 204, 209, 228, 609,
 639, 641–643
 operation of, 168
exit(), 110–111, 117
exit system call, 27, 108, 110, 167, 172
 operation of, 110–111, 172
 status, 27, 82, 111
 exported filesystem services, 245
extattrctl system call, 350
 extension header, 562, 576, 623
 external data representation, 389

F

fast filesystem, 358, 362–380
 cluster map, 377
 cylinder group, 364–365, 620
 disk structure, 363–366
 ffs_balloc(), 369, 372–374
 ffs_read(), 367–368, 377
 ffs_reallocg(), 373–374
 ffs_write(), 369
 file block allocation, 369, 372–375, 616
 file block extension, 373
 file I/O, 367–369
 fragment allocation, 374–375
 fragment-descriptor table, 374, 625
 fragmentation, 366–369
 free-space reserve, 305, 369, 380, 625
 implementation of, 363–367, 370–379
 layout policies, 370–371
 local allocation routines, 372
 organization, 363–366
 parameterization, 370

- redesign, 362–367
 - storage optimization, 366–369
- fast retransmission, TCP, 554–556
- fault rate, 138, 623
- fehdir* system call, 125, 620
- fehflags* system call, 303, 350
- fchmod* system call, 40, 350
- fchown* system call, 40, 350
- fcntl* system call, 8, 228–230, 426, 621
- fdesc** filesystem, 259
- Federal Information Processing Standard, 8
- fetch policy, 138, 623
- FFS. See *fast filesystem*
- fhopen* system call, 350
- fifo, 33, 226
- file, 33, 307, 623
 - access validation, 67
 - append-only, 304
 - control, filesystem, 296
 - deactivation, 246
 - descriptor locking, 231–233
 - executable, 61
 - flags, 303–304
 - handle, NFS, 389, 623
 - hole in, 41, 626
 - I/O, FFS, 367–369
 - immutable, 304
 - interpretation, filesystem, 296
 - management, filesystem, 296
 - mapping, 170–171
 - offset, 34, 227, 624
 - permission bits, 67
 - reclaim, 246–247
 - size distribution, 366
- file block
 - allocation, FFS, 369, 372–375, 616
 - locality of reference, 371
 - reading, 367
 - writing, 368–369
- file entry, 227–228, 623
 - flag, 228, 230
 - handling during *fork* system call, 228
 - implementation of, 228
 - object oriented, 227, 230
 - operations, 227
- file locking, 228, 231–233, 319–324
 - implementation of, 232–233, 321–324
 - NFS, 388
 - semantics of, 319–321
- file structure, 226, 451, 624
- file-table flag, 426
- filename, 38–39, 624
 - cache, 247–248
 - negative caching of, 248
 - whiteout, 257
- filestore
 - abstraction, 359–362
 - attribute update, 358
 - contents update, 358
 - creation and deletion, 358
 - implementation of, 359–362
 - operations, 358–359
 - overview, 41–42
 - size update, 359
- filesystem, 215, 624
 - 3BSD, 363, 366, 370–371
 - 4.2BSD, 363
 - 4.3BSD, 243
 - 4.4BSD, 243, 376
 - access control, 41
 - attribute manipulation, 296
 - buffering, 360–362
 - CD9660, 259
 - device, 269, 271
 - distributed, 40
 - fdesc**, 259
 - file control, 296
 - file interpretation, 296
 - file management, 296
 - independent services, 246–252
 - initialization, 608
 - ISO, 259
 - kernfs**, 259
 - layer, 255–256
 - links, 313–315
 - name creation, 295
 - name deletion, 295
 - name lookup, 310–312
 - name translation, 39, 313
 - naming, 307–315
 - NT, 40
 - nullfs**, 255–256
 - operations, 295–297
 - operator, *valloc*, 358
 - operator, *vfree*, 358
 - operator, *vget*, 358
 - overview, 38–42
 - portal**, 245, 259

- /proc**, 37, 130–131, 259, 638
 - proefs**, 259
 - quotas, 8, 315–319
 - snapshot, 349–358
 - stackable, 253–259
 - support for multiple, 37
 - umapfs**, 256, 400
 - union**, 256–258
 - see also *buffer cache*, *quotas*
 - find**, 348
 - First Edition UNIX, 79
 - first-level bootstrap, 225
 - flags, file, 303–304
 - floating point in the kernel, use of, 537
 - flock* system call, 388
 - flow control in TCP, 528
 - foreground process, 121–122, 420, 616, 624
 - fork* system call, 4, 27, 34, 41, 73, 79,
 - 84–85, 90, 108, 110, 119–120,
 - 130–131, 164–167, 174, 190, 204, 206,
 - 209, 228, 618, 624, 635, 637, 639
 - deadlock avoidance during, 166
 - file entry handling during, 228
 - implementation of, 164–166
 - implementation issues, 166–167
 - see also *process creation*
 - forward, 476, 624
 - forward-mapped page table, 197, 624
 - forwarding mechanism, 495
 - 4.0BSD, 6–8, 363
 - 4.1BSD, 6, 45
 - 4.2BSD, xx, xxvii, 6–8, 30, 34, 36, 40–41,
 - 44–46, 63, 141, 219, 231–232, 235,
 - 281, 385, 416, 435, 508, 513, 516, 525
 - filesystem, 363
 - IPC design, 8
 - virtual-memory interface, 7
 - 4.3BSD, xix, xx, xxvii, 6–8, 30, 63, 189,
 - 209–210, 231, 513, 517, 612
 - filesystem, 243
 - Reno release, 6–7
 - Tahoe release, 6–7, 9
 - 4.4BSD, xix, xxvii, 6–7, 13, 173, 513, 612,
 - 614
 - autoconfiguration, 281
 - buffer cache, 202
 - copy object, 163
 - filesystem, 243, 376
 - Lite, xx, 7, 13
 - mbuf design, 447
 - NFS, 386–387
 - page replacement, 188, 191
 - scheduler, 99–100
 - stackable filesystem, 253–254
 - supported architectures, 6
 - swap out, 195
 - swap pager, 178
 - virtual memory, 31
 - fragmentation, FFS, 366–369
 - free()*, 32, 147, 209
 - free page list, 189
 - freeblks* structure, 331, 334, 341, 343–344
 - FreeBSD
 - as a real-time system, 81, 105, 158–159
 - development model, 14–17
 - goals, 17
 - IPC design, 436, 441
 - kernel, division of software in, 25
 - portability of, 23
 - freefile* structure, 334, 342–344
 - freefrag* structure, 334–335
 - fsck**, 220, 225, 328, 347, 356–357, 363, 366,
 - 607–608
 - background, 356–357
 - dependencies, soft updates, 347
 - fstat* system call, 40, 487
 - fsync* dependencies, soft updates, 344–345
 - fsync* system call, 159, 242, 249, 260,
 - 300–301, 327–328, 338, 344–345, 350,
 - 362, 369, 376, 380, 402
 - fsync* vnode operator, 359
 - ftp**, 125
 - fruncate* system call, 350
 - futimes* system call, 350
- ## G
- gateway, 494, 625
 - handling, 496–497
 - gdb**, 129, 611
 - g_down*, 50, 274–275
 - generation number, 389, 625
 - GENIE operating system, 4
 - GEOM. See *geometry layer*
 - geometry layer, 24, 38, 264, 270–281, 286,
 - 292
 - operation, 274–275
 - topology, 270–276

getattr vnode operator, 296
getblk(), 251–252
getc(), 421–422, 427
getdirentries system call, 309
getfsstat system call, 245
gethostbyname(), 564
gethostbyname2(), 565
gethostbyname2 library call definition, 565
getlogin system call, 609
getnewbuf(), 251
getnewvnode(), 246, 248
getpeername system call, 441
getrusage system call, 71
getsockname system call, 441
getsockopt system call, 441, 484, 488
gettimeofday system call, 65–66
getty, 607–609
 GID. See *group identifier*
 global page-replacement algorithm, 188, 625
 global vnode table, benefit of, 246
 goals, FreeBSD, 17
 Greenwich time. See *Universal Coordinated Time*
 group identifier, 66–69, 73, 128, 256, 400, 622, 625, 627, 639, 641–643
 use in file-access validation, 67
gsignal(), 114
g_up, 50, 274–275, 281

H

half-open connection, 529
handle_written_inodeblock(), 334
 hard limit, 72, 625
 hard link, 313–314, 625
hardclock(), 58–60, 71, 103
hardware_cache_fetch, 185
 Harris, Guy, 9
 HBA. See *host bus adapter*
 header prediction, TCP, 543, 626
 heap, 63, 626
 high watermark on, 626
 socket, 452, 459, 505
 terminal, 424
 history of
 job control, 7
 process management, 79
 remote filesystems, 385–387
 UNIX, 3–7
 home directory, 39, 626

hop-by-hop option, 563
 hop limit, 562, 626
 host bus adapter, 266, 277
 host cache metrics, TCP, 539
 host identifier, 69
 host name, 69
 host unreachable, 556
 message, 556, 626

I

I/O, 627
 asynchronous, 229–231
 nonblocking, 229, 234, 423, 429, 457, 459, 462, 634
 physical, 220–221
 queueing, 217–218
 redirection, 34, 628
 request, ATA, 280–281
 request, CAM, 277–279
 scatter/gather, 36–37, 46, 458, 642
 signal driven, 229, 234, 644
 system design, 33–37
 types of kernel, 215–217
 I/O buffer, 223
 I/O stream, 33, 628
 I/O vector, 239–240
 ICMP. See *Internet control message protocol*
icmp_error(), 558
icmp_input(), 557
 idempotent, 389, 626
 identification, device, 284–286
 identify, 283
 idle
 loop, 104, 626
 process, 50, 601
 queue, 626
 swap time, 195
 IEEE. See *Institute of Electrical and Electronic Engineers*
 IETF. See *Internet Engineering Task Force*
ifaddr structure, 479, 483, 499
ifconfig, 430
if_data structure, 481
if_done(), 484
ifnet structure, 479, 483
if_output(), 483, 567, 569
if_start(), 484
 IGMP. See *Internet group management protocol*

- ignored signal, 28
- imgact*, 61
- immutable file, 304
- implementation of
 - ARP, 509–510
 - buffer cache, 251–252
 - dup* system call, 229
 - FFS, 363–367, 370–379
 - file entry, 228
 - file locking, 232–233, 321–324
 - filestore, 359–362
 - fork* system call, 164–166
 - ioctl* system call, 230
 - jail, 125–129
 - kernel malloc, 148–149
 - munmap* system call, 170–171
 - NFS, 393–397
 - pipe, 34
 - pmap_enter*(), 204–205
 - pmap_remove*(), 205–206
 - quotas, 315–319
 - select* system call, 236–239
 - sleep*(), 86–87, 90–91
 - sysctl* system call, 612–614
 - uiomove*(), 239–240
 - wakeup*(), 92–93
- inactive page list, 188, 207
- inactive, reclaim from, 191
- inactive* vnode operator, 246–247, 296, 307
- inbound, 475, 627
- indirdep* structure, 336–337
- indirect block dependencies, soft updates, 336–338
- indirect route, 496
- init**, 27, 49, 86, 129, 190, 304, 601, 606–609, 627, 652
- initial sequence number, 528, 627, 643
- initialization
 - filesystem, 608
 - kernel, 596–597
 - user-level system, 607–609
 - virtual memory, 201–204, 208
 - see also *bootstrapping*
- initiate_write_inodeblock*(), 334
- inode, 240, 360, 380–381, 627
 - allocation, 299
 - cache, 306–307
 - contents, 297
 - definition, 297–305
 - dependencies, soft updates, 332–334
 - locality of reference, 370
 - management, 306–307
 - number, 257, 301, 305–308, 310, 332, 338, 343, 352, 389–390, 621
- inode wait*, 334–335, 341–342
- inodedep* structure, 332–335, 337–339, 344
- inpcb* structure, 518, 538
- in_pcballoc*(), 519
- in_pcbbind*(), 519
- in_pcbconnect*(), 519, 538
- in_pcbdetach*(), 522
- in_pcblookup*(), operation of, 521
- insecure mode, 304
- Institute of Electrical and Electronic Engineers, 7, 267, 636
- integrity-check value, 573, 627
- interactive program, 81, 627
- Interdata 8/32, 5
- interface
 - addresses, network, 479–481
 - buffer cache, 249
 - capabilities, network, 481–483
 - character device, 215–216, 219, 221–223, 415, 618
 - disk device, 223
 - line switch, 415, 630
 - mmap* system call, 157–159
 - network, 479–484
 - output, network, 483–484
 - pager, 173–180
 - protocol–network, 491–494
 - protocol–protocol, 488–491
 - socket-to-protocol, 484–488
 - virtual-filesystem, 240–245
- International Organization for Standardization, 630
 - filesystem, 259
 - model, 474, 514
 - protocol suite, 461, 508, 513
- Internet addresses
 - broadcast, 516–517
 - multicast, 517
 - packet demultiplexing, 517
 - structure, 454
- Internet control message protocol, 491, 503, 506, 514–515, 521, 526, 552, 556–558, 565, 567, 569, 572, 626–627
 - interaction with routing, 557
 - port unreachable message, 521
- Internet domain, 6, 44

- Internet Engineering Task Force, 387, 563
 - Internet group management protocol, 517
 - Internet key exchange, 574, 627
 - Internet protocol, xx, 3, 6, 45, 126–129, 398–399, 430, 476, 506, 514–528, 542, 547, 556–558, 584–585, 628
 - fragmentation, 514, 522, 524–525
 - handling of broadcast message, 523
 - input processing, 524–526
 - multicast router, 526
 - options, 522
 - output processing, 523–524
 - packet demultiplexing, 518
 - packet forwarding, 525–526, 558
 - protocol header, 522
 - pseudo-header, 519, 521, 542
 - responsibilities of, 522
 - version 4 addresses, 515
 - Internet service providers, 515–516, 559
 - interpreter, 61, 627
 - interprocess communication, 6, 22, 34, 36, 44–45, 71, 435–468, 470, 604, 627–628
 - connection setup, 455–457
 - data structures, 449–455
 - data transfer, 457–463
 - design, 4.2BSD, 8
 - design, FreeBSD, 436, 441
 - layers, 441–442
 - local interprocess communication, 464–468
 - memory management in, 443–449
 - message queue, 435, 467–468, 470
 - model of, 435–441
 - receiving data, 460–463
 - reliable delivery, 459
 - semaphores, 465–467
 - shared memory, 156–164, 468
 - socket shutdown, 463–464
 - startup, 604–606
 - transmitting data, 458–460
 - interprocessor interrupt, 107
 - interrupt, 628
 - device, 56–57, 628
 - request, 265
 - interrupt handling, 56–57
 - clock, 57–59
 - device driver, 218
 - interrupted system call, 54–55
 - interruptible *sleep()*, 86, 115
 - interval time, 66
 - inverted page table, 197
 - involuntary context switching, 89, 104
 - ioctl, character device, 222
 - ioctl* system call, 35, 122, 227, 230, 415–418, 420, 428–429, 462, 479, 483, 487, 619
 - implementation of, 230
 - ioctl* vnode operator, 296
 - iovec* structure, 239–240, 628, 642, 650
 - IP. See *Internet protocol*
 - ip6_output()*, 567
 - IPC. See *interprocess communication*
 - IPC_NOWAIT, 467
 - ip_forward()*, 580
 - IPI. See *interprocessor interrupt*
 - ip_input()*, 580
 - ipintr()*, operation of, 524–526
 - ip_output()*, 520–521, 523, 526, 547, 581
 - operation of, 523–524
 - IPSec, 479, 513, 523, 525–526, 560, 569–572, 574, 577, 579–581, 583, 585–586, 627–628, 642, 649
 - authentication header, 572
 - encapsulating security protocol, 574
 - implementation, 579–581
 - overview, 570–571
 - ipsec4_process_packet()*, 581
 - ipsec_common_input()*, 580
 - IPv6, 44–45, 454, 513, 558–569, 572–573, 577, 579, 584–586, 627–628, 633, 650
 - addresses, 560–561
 - autoconfiguration, 565–569
 - introduction, 558–560
 - packet formats, 562–563
 - socket API changes, 563–565
 - IRQ. See *interrupt request*
 - ISA bus, 24–25, 266–268, 283–285, 292
 - ISA. See *ISA bus*
 - ISO. See *International Organization for Standardization*
 - ISP. See *Internet service providers*
 - issignal()*, operation of, 117
 - ITS operating system, 7
- ## J
- jail, 123–129
 - implementation of, 125–129

- motivation, 123–125
 - network address, 126–129
 - jail* system call, 126–127
 - jailkill* system call, 129
 - job*, 69, 122
 - job control, 29, 122–123, 629
 - history of, 7
 - signals in FreeBSD, 28
 - terminal driver support for, 420, 424, 427
 - use of process group, 29
 - Joy, William, 6
- K**
- KAME, 45, 558, 570
 - keepalive packet, 536, 629
 - keepalive timer, 536, 629
 - Kerberos authentication, 395, 400–401
 - /kern**, 259
 - KERNCONF, 609
 - kernel, 22, 629
 - address space allocation, 146–150
 - assembly language in the, 25, 54, 104, 218, 595–598, 600–601
 - bottom half of, 51–52
 - configuration, 609–610
 - entry to, 52–53
 - I/O, types of, 215–217
 - initialization, 596–597
 - loadable modules, 598, 603–604, 629
 - loading of, 201
 - memory allocation, 32
 - memory management, 144–150
 - mode, 79, 140, 629
 - module initialization, 598–607
 - organization, 23–25
 - partitioning, reason for, 22
 - preemption, 52
 - process, 49, 629
 - resource allocation, 165–166
 - return from, 54
 - security level, 304
 - state, 80, 629
 - thread initialization, 600–601
 - top half of, 51–52
 - kernel malloc, 147–149
 - implementation of, 148–149
 - requirements, 147–148
 - kernfs** filesystem, 259
 - key, 464, 629
 - management, 574–579
 - kill character, 414, 629
 - kill* system call, 113
 - killpg* system call, 122, 637
 - kmem_alloc*(), 146
 - kmem_alloc_pageable*(), 146
 - kmem_alloc_wait*(), 146
 - kmem_free*(), 146
 - kmem_free_wakeup*(), 146
 - kmem_malloc*(), 146
 - kqueue, 226–227, 630
 - kse_create* system call, 83
 - ktrace* system call, 350
- L**
- LAN. See *local-area network*
 - Lawson, Nate, xxiv
 - layout, virtual memory, 141–142
 - lbolt*, 90, 425
 - lchmod* system call, 350
 - lchown* system call, 350
 - lease, 630
 - NFS, 394, 404–408
 - noncaching, 405–406, 634
 - obtaining an, NFS, 407
 - read-caching, 405, 639
 - write-caching, 405–406, 652
 - least recently used, 155, 318–319, 630–631
 - library, shared, 64
 - lightweight process, 82, 133
 - limits in system, 316
 - line discipline, 415–416, 423, 430, 630
 - close*(), 429–430
 - output*(), 424–425
 - PPP, 430
 - line mode, 414, 630
 - line switch interface, 415, 630
 - linesw* structure, 415, 630
 - link count dependencies, soft updates, 345–347
 - link layer, 474, 630
 - path, 479
 - link* system call, 40, 350. See also *filesystem links*
 - link* vnode operator, 295
 - Linux operating system, xix, xx, xxi, 7, 11, 17, 62, 84, 622

- LISP programming language, 6
 - listen request, 485, 630
 - listen* system call, 439, 455, 540, 630
 - definition, 439
 - Lite, 4.4BSD, xx, 7, 13
 - live **dump**, 358
 - llinfo_nd6*, 568
 - ln_hold*, 568
 - load average, 101–102, 630
 - local-area network, 548, 571, 606
 - local domain, 44, 295, 630
 - address structure, 454
 - local page-replacement algorithm, 188, 631
 - local-remote decision, 496
 - locality of reference, 139, 370–371, 631
 - lock* vnode operator, 296
 - LOCKED buffer list, 250
 - locking
 - advisory, 232, 296, 615
 - file descriptor, 231–233
 - mandatory, 232, 631
 - resources on a shared-memory multiprocessor, 93–99, 455
 - resources, deadlock avoidance when, 94–96, 465–466, 470
 - semantics of, file, 319–321
 - socket data buffer, 460
 - lockinit*(), 98
 - logical block, 360, 631
 - login**, 67–68, 304, 609
 - login name, 70
 - login shell, 22
 - long-term scheduling algorithm, 101
 - lookup* vnode operator, 244, 296
 - lost+found**, 347
 - low watermark on, 631
 - socket, 452
 - terminal, 424, 426
 - LRU. *See least recently used*
 - ls**, 370
 - lseek* system call, 34, 227, 624
 - lstat* system call, 315
 - lutimes* system call, 350
- M**
- MAC. *See mandatory access control*
 - Mach operating system, 7, 22, 31, 141, 163, 173, 177, 199, 207
 - Macklem, Rick, 393
 - m_adj*(), 448
 - magic number, 61, 631
 - main memory, 135, 631
 - major device number, 216, 631
 - make**, 81
 - malloc*(), 32, 63, 141–142, 146–147, 169, 209, 252, 478, 626
 - management information base, 613
 - mandatory access control, 301–302, 631
 - mandatory locking, 232, 631
 - mapping, 199
 - physical to virtual, 202–204
 - structure, 199, 632
 - maps, virtual memory, 145–146
 - MAP_SHARED, 248
 - marshalling of arguments, 388, 632
 - Massachusetts Institute of Technology, 4, 7
 - master boot record, 270–273
 - maxcontig*, 376–377
 - maximum segment lifetime, 530, 532, 585–586, 632–633. *See also 2MSL timer*
 - maximum-segment-size option, TCP, 530, 538
 - maximum transmission unit, 399, 499, 505, 538–540, 586
 - maximum_lease_term*, 404, 408
 - maxusers*, 447
 - mb_alloc*(), 448
 - MBR. *See master boot record*
 - mbuf, 145, 443–447, 632
 - allocation, 447–448
 - cluster, 443–449
 - data structure description, 443–446
 - design, 446–447
 - design, 4.4BSD, 447
 - storage-management algorithm, 447–448
 - utility routines, 448–449
 - m_copy*(), 547
 - m_copydata*(), 448, 547
 - m_copym*(), 448
 - memory allocation
 - buffer cache, 252
 - kernel, 32
 - memory deadlock, 165, 178–180, 194–195
 - memory management, 29–32, 135–209
 - cache design, 184–186
 - design, 30–32

- goals, 135–141
 - hardware, VAX, 30
 - in IPC, 443–449
 - kernel, 144–150
 - page-table design, 198–199
 - portability of, 30
 - system, 135, 632
 - memory-management unit, 137, 184, 197–198, 201–202, 207, 632
 - design, 196–198
 - memory overlay, 137
 - merged from current, 16
 - message queue, 44, 632
 - System V, 435, 467–468
 - metadata, 159, 248–249, 252, 258, 273, 324–328, 345, 347, 353, 378–379, 632, 650
 - metrics, route, 499, 505
 - M_EXT, 445
 - MFC. *See merged from current*
 - m_free()*, 448
 - m_freem()*, 448
 - m_get()*, 448
 - m_gethdr()*, 448
 - m_hdr* structure, 443
 - MIB. *See management information base*
 - MINIX operating system, 7
 - minor device number, 216, 632
 - mi_startup()*, 598
 - mi_switch()*, 89–91, 104–105
 - mkdir* structure, 334, 339–340
 - mkdir* system call, 40, 46, 340, 350
 - mkdir* vnode operator, 295
 - MKDIR_BODY flag, 339
 - MKDIR_PARENT flag, 339
 - mkfifo* system call, 350
 - mknod* system call, 350
 - mknod* vnode operator, 295
 - mlock* system call, 158–159, 188, 207
 - definition of, 158
 - mmap* system call, 30, 32, 64, 142, 157–158, 165, 170–171, 174, 204, 248, 632
 - definition of, 157
 - interface, 157–159
 - mmap* vnode operator, 296
 - MMU. *See memory-management unit*
 - modem control, 422–423, 633
 - ignored, 423
 - motivation for *select* system call, 233–236
 - mount**, 358, 395
 - mount options, 245
 - mount* system call, 37, 222, 253, 256, 258, 394–395, 608
 - mountd**, 394–395, 400
 - M_PKTHDR, 444
 - M_PREPEND()*, 449
 - mprotect* system call, 158, 171, 206
 - definition of, 158
 - m_pullup()*, 449, 521, 542
 - MS-DOS operating system, 387–388
 - msgrcv* system call, 467–468
 - msgsnd* system call, 467
 - MSL. *See maximum segment lifetime*
 - msleep()*. *See sleep()*
 - msg_mtx*, 467
 - msync* system call, 159, 174, 176–177
 - definition of, 159
 - mtod()*, 448
 - MTU. *See maximum transmission unit*
 - mtx_destroy()*, 97
 - mtx_init()*, 96–97
 - mtx_lock()*, 97
 - mtx_trylock()*, 97
 - mtx_unlock()*, 97
 - multicast, 481
 - Internet addresses, 517
 - message, 521
 - router, IP, 526
 - Multics operating system, 4, 7
 - multilevel feedback queue, 100, 633
 - multiprocessor
 - locking resources on a shared-memory, 93–99, 455
 - virtual memory for a shared-memory, 30
 - multiprogramming, 79–80
 - munlock* system call, 159
 - definition of, 159
 - munmap* system call, 158, 161, 165, 170, 205
 - definition of, 158
 - implementation of, 170–171
 - mutex, network thread, 494
 - mutex, spin, 94–97, 645
- N**
- Nagle, John, 548

- name
 - creation, filesystem, 295
 - deletion, filesystem, 295
 - login, 70
 - lookup, filesystem, 310–312
 - translation, filesystem, 39, 313
 - named object, 154
 - naming
 - filesystem, 307–315
 - shared memory, 157–158
 - National Bureau of Standards, 7
 - nd6_na_input()*, 569
 - nd6_output()*, 567–569
 - nd6_timer()*, 569
 - nd_input()*, 567
 - negative caching of filename, 248
 - neighbor discovery, 565, 567, 633
 - Net1 release, 7
 - Net2 release, 7
 - NetBSD, xix, xx, xxiii, 3, 10–11, 13–14, 244, 358
 - netmask, 515
 - network
 - additions in FreeBSD, 45
 - address, jail, 126–129
 - architecture, 473, 633
 - buffering, 505–506
 - byte order, 515, 633
 - congestion control, 505–506
 - data flow, 475–476
 - deadlock, 233–234
 - design, 45
 - layer, 474, 633
 - mask, 633
 - protocol capabilities, 478–479
 - queue limiting, 506
 - thread*, 524
 - thread mutex, 494
 - time protocol, 603
 - time synchronization, 65–66
 - timer, 59, 477
 - Network Disk Filesystem, 386
 - Network Filesystem, 42, 241, 247, 249, 255–256, 258, 296–297, 385–410, 452, 606, 617, 623, 625, 634, 639, 652
 - 4.4BSD, 386–387
 - asynchronous writing, 402
 - client-server interaction, 397–398
 - crash recovery, 408–409
 - daemons, 394–397
 - delayed writing, 402
 - design, 387–388
 - file handle, 389, 623
 - file locking, 388
 - hard mount, 397
 - implementation of, 393–397
 - interruptible mount, 398
 - lease, 394, 404–408
 - lease, obtaining an, 407
 - overview, 42
 - protocol, 391–393
 - recovery storm, 409, 639
 - RPC transport, 398–400
 - security issues, 400–401
 - soft mount, 398
 - structure, 388–401
 - network interface, 479–484
 - addresses, 479–481
 - capabilities, 481–483
 - layer, 474, 633
 - output, 483–484
 - newblk* structure, 332
 - newbus, 38, 268, 281, 633
 - newfs**, 305, 364, 366
 - next header, 563
 - nextc()*, 422
 - NFS. See *Network Filesystem*
 - nfsd**, 394–395, 397, 399–401, 409
 - nfsiod**, 396–397
 - nfsvc* system call, 395–396
 - nice*, 28, 70–71, 196, 633–634, 638, 642
 - Ninth Edition UNIX, 5
 - nonblocking I/O, 229, 234, 423, 429, 457, 459, 462, 634
 - noncaching lease, 405–406, 634
 - nonlocal goto, 634
 - Not-Quite Network Filesystem, 394, 396, 404, 406–410, 630
 - Novell, 6, 8
 - NQNFS. See *Not-Quite Network Filesystem*
 - NT filesystem, 40
 - NTP. See *network time protocol*
 - null modem connection, 423
 - nullfs** filesystem, 255–256
- O**
- O_ASYNC flag, 229, 347–348

- object, 141
 - oriented file entry, 227, 230
 - shadow, 143, 154, 160–164, 643
 - virtual memory, 153–156, 651
 - obtaining BSD, xxiii
 - octet, 515, 634
 - Olson, Arthur, 9
 - open-file entry*, 306
 - open source BSD, 9–13
 - open* system call, 33, 35, 41, 174, 222, 227, 253, 296–297, 306, 314–315, 350, 416, 423, 441, 465, 621
 - open* vnode operator, 296
 - OpenBSD, xix, xx, xxiv, 3, 11, 14, 570, 578, 581
 - opendir()*, 309
 - operation of */dev*, 268–269
 - operations
 - filestore, 358–359
 - filesystem, 295–297
 - terminal, 423–430
 - optimal replacement policy, 138, 634
 - organization, FFS, 363–366
 - orphaned process group, 123, 634
 - OS/X operating system, Apple, 3, 300, 309
 - OSF/1, 84
 - OSI. See *International Organization for Standardization*
 - out-of-band data, 460–462, 487, 508–509, 634
 - receipt of, 462
 - transmission of, 458
 - overlay, 25, 634
 - overview, soft updates, 324–325
- P**
- packet
 - filter, 477, 482
 - forwarding, IP, 525–526, 558
 - queue, 493–494
 - reception, 493–494
 - transmission, 491–492
 - packet demultiplexing
 - Internet addresses, 517
 - IP, 518
 - page cache, 216, 220
 - page clustering, 174, 184, 191, 209, 618
 - page coloring, 186–187
 - page fault, 137, 624, 635, 639
 - page lists, 188–189
 - active, 188
 - cache, 189
 - free, 189
 - inactive, 188, 207
 - wired, 188
 - page replacement, 6, 138–139, 187–194
 - 4.4BSD, 188, 191
 - criterion for, 187–189
 - in the VMS operating system, 188
 - page table, 198
 - forward-mapped, 197, 624
 - pages, 198, 635
 - page-table entry, 197–199, 203, 205–206, 208, 635, 638
 - page usage, 207–208
 - page, wired, 146, 176, 199–200, 202, 207, 651
 - pagedaemon*, 50, 82, 146, 150, 154, 173–174, 176–177, 179–180, 188–195, 207–209, 350, 606, 620, 629, 635
 - operation of the, 190–194
 - pagedep* structure, 338–339, 341, 343–344
 - pagein()*, 635
 - operation of, 180–184
 - pageout daemon. See *pagedaemon*
 - pageout in progress, 179
 - pager, 153–155, 635
 - definition of, 173–174
 - device, 154, 175–176
 - interface, 173–180
 - physical-memory, 176–177
 - swap, 154, 177–180
 - vnode, 154, 174–175
 - pagezero*, 50, 606
 - paging, 6, 30, 63, 137–138, 140, 152–153, 155–156, 180–187, 621, 635
 - parameters, 189–190
 - systems, characteristics of, 138
 - panic, 610, 635. See also *system crash*
 - parent directory, 39
 - parent process, 26, 85, 108, 635
 - partition. See *disk partition*
 - path MTU discovery, 499, 539, 586, 635
 - pathname, 39, 635
 - translation, 244–245
 - paths through network node, 475
 - PC. See *personal computer*

- PCI. See *peripheral component interconnect*
- PCMCIA, 265
- PDP-11, 5, 54, 79
- PDP-7, 3, 79
- performance
 - snapshot, 355–356
 - soft updates, 347–349
 - see also *system performance*
- peripheral component interconnect, 24–25, 264–265, 267–268, 284–289, 292
- permanent kernel module, 598, 636
- persist timer, 536, 636
- personal computer, 52–54, 57, 59, 65, 365
 - architecture, 264–266
 - stack growth on, 63
- pfctl*input(), 557
- PF_KEY
 - address extension, 577
 - association extension, 576
 - base header, 576
- PF_KEY_V2, 574
- pgo_alloc*(), 174
- pgo_dealloc*(), 174
- pgo_getpage*(), 177
- pgo_getpages*(), 174–176
- pgo_haspage*(), 174, 177
- pgo_init*(), 174
- pgo_putpages*(), 174–177, 179
- physical block, 360, 636
- physical I/O, 220–221
 - algorithm for, 221
- physical mapping, 199
- physical-memory pager, 176–177
- physical to virtual mapping, 202–204
- physio*(), 220, 222
- PID. See *process identifier*
- ping**, 129, 506, 557
- pipe, 33, 35, 226, 436, 636
 - implementation of, 34
 - system call, 33, 35, 227, 621
- pipeline, 29, 35, 636
- placement policy, 138, 636
- Plan 9, 5
- pmap, 199–209, 636
 - functions, 201
 - initialization, 202
 - module, 143, 199–201, 209
 - structure, 143, 636
- pmap_bootstrap*(), 201–202
- pmap_change_wiring*(), 201, 207
- pmap_clear_modify*(), 201, 207
- pmap_clear_ptes*(), 207
- pmap_clear_reference*(), 201, 207
- pmap_copy_page*(), 201, 208
- pmap_enter*(), 201, 204–206
 - implementation of, 204–205
- pmap_growkernel*(), 201–202
- pmap_init*(), 201–202
- pmap_is_modified*(), 201, 208
- pmap_page_protect*(), 201, 206–207
- pmap_pinit*(), 201, 209
- pmap_protect*(), 201, 206–207
- pmap_qenter*(), 201, 206
- pmap_qremove*(), 201, 206
- pmap_release*(), 201, 209
- pmap_remove*(), 201, 205–208
 - implementation of, 205–206
- pmap_remove_all*(), 207
- pmap_ts_referenced*(), 201, 207–208
- pmap_zero_page*(), 201, 208
- point-to-point protocol, 430, 636–637
- poll interface, System V, 235
- poll* system call, 222, 234–236, 416, 636
- poll* vnode operator, 296
- pollfd* structure, 235
- polling I/O, 234, 636
- portability of
 - FreeBSD, 23
 - memory management, 30
 - Seventh Edition UNIX, 5
- portable operating system interface, xx, xx, 7–8, 69, 123, 156, 230–232, 319, 328, 416–417, 636–637
- portal** filesystem, 245, 259
- portmap**, 394–395
- POSIX. See *portable operating system interface*
- postsig*(), 115, 117–118
 - operation of, 117–118
- ppp**, 430
- PPP. See *point-to-point protocol*
- pr_ctlinput*(), 478, 491, 502, 521, 557
- pr_ctloutput*(), 478, 484, 488, 521
- pr_drain*(), 477
- preemption
 - kernel, 52
 - thread, 105
- prefetching, 637

- prefix option, 566
- prepaging, 138, 637
- pr_fasttimo()*, 477
- pr_input()*, 478, 489–490
- prison* structure, 127
- private mapping, 157, 160, 637
- private memory, 160–161, 163–164
- probe, 283, 637
- /proc** filesystem, 37, 130–131, 259, 638
- process, 26, 79, 637
 - context, 26, 637
 - creation, 108–110, 164–167
 - debugging, 116, 129–131
 - flags, 130
 - kernel, 49, 629
 - lightweight, 82, 133
 - open-file table, 306, 637
 - profiling, 56, 66
 - resource accounting, 58–59, 73–74, 111
 - scheduling, 50, 59–60, 64, 81–82
 - state, change of, 111, 116–117, 130
 - state organization, 82–88
 - structure, 51–52, 80, 84–87, 89, 638
 - termination, 110–111, 172
 - virtual address space, 151
 - virtual memory duplication, 166–167
 - virtual memory resources, 151–156
 - virtual time, 66
- process group, 29, 69–70, 119–120, 122, 637
 - association with, socket, 122, 451
 - hierarchy, 85
 - identifier, 119, 229, 451, 637
 - job-control use of, 29
 - leader, 119
 - orphaned, 123, 634
 - terminal, 122, 420, 426–427, 429
- process identifier, 26–27, 70, 82–84, 108–110, 119–120, 132, 164, 420, 504, 636–637
 - allocation, 109
- process management, 26–29, 61–65, 79–131
 - history of, 79
- process priority, 28, 55, 70–71, 638
 - calculation of, 59
- processor affinity, 105–106, 150, 637
- processor group, 107, 638
- processor status longword, 52–54
- procf**s filesystem, 259
- profclock()*, 58, 66
- profil* system call, 75
- profiling
 - process, 56, 66
 - timer, 58, 66
- program relocation, 596, 640
- programming language
 - B, 4
 - BCPL, 4
 - C, 3–4, 26, 54
 - LISP, 6
- protection, virtual memory map, 206–207
- protocol, 44, 619
 - buffering policy, 505
 - capabilities, network, 478–479
 - control block, 518–519
 - control-output routine, 488
 - network interface, 491–494
 - NFS, 391–393
 - protocol interface, 488–491
 - switch, 450
 - switch structure, 477, 638
- protocol family, 438, 473, 638
- protosw* structure, 450
- pr_output()*, 478, 489
- pr_pfil()*, 477
- pru_abort()*, 487
- pru_accept()*, 486
- pru_attach()*, 484–485
- pru_bind()*, 484–485
- pru_connect()*, 484–485
- pru_connect2()*, 487
- pru_control()*, 487
- pru_detach()*, 485–486
- pru_disconnect()*, 486
- pru_fasttimo()*, 487
- pru_listen()*, 485
- pru_peeraddr()*, 487
- pru_rcvd()*, 486
- pru_rcvoob()*, 487
- pru_send()*, 484, 486
- pru_sense()*, 487
- pru_shutdown()*, 486
- pru_slowtimo()*, 487
- pru_sockaddr()*, 487
- pr_usrreqs()*, 478, 484–485, 488
- ps**, 83–84, 87
- pseudo-header, IP, 519, 521, 542

- pseudo-terminal, xxiii, 21, 23, 29, 43, 128, 237–238, 247, 269, 413–415, 418, 423–427, 431, 618–619, 632, 638, 644, 649
 - psignal()*, 114, 116–117
 - operation of, 116–117
 - PSL. *See processor status longword*
 - ps_strings* structure, 64
 - PTE. *See page-table entry*
 - ptrace* system call, 93, 129–131
 - public key encryption, 638
 - pure demand-paging, 138, 638
 - push migration, 638
 - putc()*, 422
 - pv_entry* structure, 200, 202–208, 210
- Q**
- q_to_b()*, 422
 - queue limiting, network, 506
 - quotacheck**, 319
 - quotactl* system call, 350
 - quota.group**, 316
 - quotas
 - contribution of, 8
 - format of record, 316
 - implementation of, 315–319
 - limits, 316
 - quota.user**, 316
- R**
- race condition, 119–120, 156, 183, 233, 346, 638
 - radix search trie, 499
 - RAID. *See redundant array of inexpensive disks*
 - range lock, System V, 231
 - rapid connection reuse, 544, 639
 - raw device, 220–221
 - interface, 219, 639
 - raw mode, 43, 414
 - raw socket, 35, 473, 506–508, 515, 557, 639
 - control block, 506–507
 - input processing, 507–508
 - output processing, 508
 - read-caching lease, 405, 639
 - read* system call, 31, 33, 36–37, 44, 131, 227, 236, 240, 248, 253, 260, 415, 426–428, 441, 458, 624, 634, 638, 647
 - read* vnode operator, 358–359
 - READ_10, 278
 - readdir()*, 309
 - readdir* vnode operator, 296
 - readlink* vnode operator, 296
 - readv* system call, 36–37, 239, 628
 - real GID. *See real group identifier*
 - real group identifier, 67, 639
 - real-time clock, 50
 - real-time system, FreeBSD as a, 81, 105, 158–159
 - real-time timer, 59, 66
 - real UID. *See real user identifier*
 - real user identifier, 67, 639
 - reboot* system call, 610, 614
 - operation of, 610–611
 - receive window, 532, 639, 645
 - reclaim from inactive, 191
 - reclaim* vnode operator, 246–247, 296, 307
 - reclamation dependencies, soft updates, 342–344
 - recovery storm, NFS, 409, 639
 - recv* system call, 36–37
 - recvfrom* system call, 36–37, 458, 564
 - recvit()*, 458
 - recvmsg* system call, 36, 458, 463
 - data structures for, 440
 - red zone, 32, 147, 640
 - redundant array of inexpensive disks, 264, 267, 270, 272
 - reference string, 138, 640
 - region, 151, 640
 - relative pathname, 39, 636, 640
 - reliably-delivered-message socket, 511, 640
 - Remote Filesystem, 386
 - remote filesystem performance, 401–404
 - remote filesystems, history of, 385–387
 - remote procedure call, 388–389, 391–403, 405–410, 640–641
 - transport, NFS, 398–400
 - remove* vnode operator, 295
 - rename* system call, 40, 350, 391
 - addition of, 40
 - rename* vnode operator, 295
 - replacement policy, 138, 640
 - request for comments, 541, 558, 560, 570
 - resetpriority()*, 103, 105
 - resident-set size, 189, 640
 - resource
 - accounting, process, 58–59, 73–74, 111

- autoconfiguration, 289–292
 - limit, 26, 70–72
 - process virtual memory, 151–156
 - sharing, 93–99
 - utilization, 71
 - retransmit timer, 535, 540, 640
 - return from kernel, 54
 - return from system call, 55–56
 - reverse-mapped page table, 197, 628, 640
 - revocation of controlling terminal, 247
 - revoke* system call, 247, 350, 420, 429, 608
 - rewinddir()*, 309
 - RFC. See *request for comments*
 - rfork* system call, 83–84, 108
 - RFS. See *Remote Filesystem*
 - rip_input()*, 557
 - Ritchie, Dennis, 3–4, 7
 - rlimit* structure, 84
 - rm**, 346
 - rmdir* system call, 40, 340, 345, 350
 - rmdir* vnode operator, 296
 - root directory, 38–39, 640
 - root filesystem, 40, 593, 641
 - root user, 66, 647
 - root0*, 268, 284–289
 - round robin, 100, 641
 - round-trip time, 399, 401–402, 537, 584
 - RPC timeout, 399
 - TCP estimation of, 537–538
 - roundrobin()*, 102, 105
 - route, cloning, 498, 509, 618
 - route metrics, 499, 505
 - routed**, 503
 - router, 476, 494, 641
 - advertisement, 565
 - entry, 566
 - IP multicast, 526
 - solicitation, 566, 641
 - routing, 494–505
 - daemon, 503, 620, 641
 - information protocol, 503
 - interaction with ICMP, 557
 - interface, 504–505
 - lookup, 499–501
 - mechanism, 496–503, 641
 - policy, 503, 641
 - redirect, 502, 641
 - socket, 504
 - tables, 496–503
 - types of, 496
 - RPC. See *remote procedure call*
 - rpc.lockd**, 392, 394, 396
 - rpc.statd**, 394, 396
 - RS-232 serial line, 413, 422, 648
 - rtalloc()*, 502
 - rtalloc_ign()*, 524
 - rtentry*, 568
 - structure, 497, 523
 - rtfree*, 502
 - rtprio* system call, 86
 - rtredirect()*, 503, 557
 - RTT. See *round-trip time*
 - run queue, 85, 100, 641
 - management of, 103–105
 - runq_add()*, 104
 - runq_choose()*, 104
 - operation of, 104
 - runq_remove()*, 104
- ## S
- SA. See *security association*
 - Samba, 386
 - Santa Cruz Operation, xix, 6
 - savecore**, 611
 - saved GID, 69, 641
 - saved UID, 68–69, 642
 - sbappendstream()*, 545–546
 - sblock()*, 460
 - sbrk* system call, 63, 141–142, 165, 169, 626
 - sbunlock()*, 460
 - SC22 WG15, 8
 - scatter/gather I/O, 36–37, 46, 458, 642
 - sched_balance()*, 107
 - schedcpu()*, 102, 105
 - sched_lock*, 91
 - schednetisr()*, 494
 - scheduler()*, 195
 - scheduler, 4.4BSD, 99–100
 - scheduling, 80, 642
 - class, 86, 642
 - long-term algorithm, 101
 - parameters, 26
 - priority, 86, 642
 - process, 50, 59–60, 64, 81–82
 - short-term algorithm, 101
 - thread, 93, 100–108
 - SCO. See *Santa Cruz Operation*
 - SCSI. See *small computer system interface*
 - secondary storage, 135, 642

- secure mode, 304
- security, 569–583
 - association, 570–571, 573–578, 580, 641–642
 - association, transport mode, 571
 - association, tunnel mode, 571
 - introduction, 569–570
 - issues, NFS, 400–401
 - level, kernel, 304
 - parameter index, 570–573, 577, 580, 642, 645
 - protocols, 572–574
 - protocols implementation, 579–581
 - system, 123–129
- seekdir()*, 309
- segment, 136, 528, 642
 - bss, 61, 617
 - data, 29, 61, 63, 169, 620
 - stack, 29, 61, 169, 645
 - text, 29, 61, 63, 648
- select* system call, 222, 234–238, 260, 452, 540, 636
 - device driver code for, 238
 - device driver support for, 222, 236–237
 - implementation of, 236–239
 - motivation for, 233–236
- selinfo* structure, 237–239
- selrecord()*, 237–239
- seltrue()*, 222
- selwait*, 237–238
- selwakeup()*, 236–238, 426
- semaphores, 44, 435, 643
 - System V, 464–466, 605
 - virtual memory, 156–157
- semctl* system call, 470
- semget* system call, 466, 470
- semop* system call, 466, 470
- send* system call, 36–37, 44, 451, 576
- send window, 532, 643
- sendfile* system call, 32
- sendit()*, 458
- sendmsg* system call, 36, 458, 484, 519
 - data structures for, 440
- sendsig()*, 118
- sendto* system call, 36–37, 458, 484, 519, 564
- sense request, 487, 643
- sequence numbers, TCP, 528
- sequence space, 528, 643
- sequence variables, TCP, 531–534
- sequenced packet socket, 437, 643
- server message block, 386
- server process, 42, 643
- session, 29, 69–70, 119–122, 420, 643
 - leader, 120, 643
- set-group-identifier program, 67, 643
- set-user-identifier program, 67, 643
- setattr* vnode operator, 296
- setuid* system call, 69
- setlogin* system call, 609
- setpgid* system call, 119–120
- setpriority* system call, 637
- setrunnable()*, 89, 103, 105, 116
- setsid* system call, 120
- setsockopt* system call, 441, 464, 484, 488, 517, 520, 549, 619
- settimeofday* system call, 65
- Seventh Edition UNIX, 5
 - portability of, 5
- sh** shell, 62, 607
- shadow object, 143, 154, 160–164, 643
 - chain, 161–163
 - collapse, 161–163
- shared library, 64
- shared mapping, 157, 643
- shared memory, 44, 156–164, 435, 644
 - naming, 157–158
 - System V, 157, 176, 468
- shared text segment, 6
- sharing, resource, 93–99
- shell, 644
 - cs**, 122
 - login, 22
 - sh**, 62, 607
- shmat* system call, 468
- shmdt* system call, 468, 470
- shm* system call, 157, 468
- shmget* system call, 468
- short-term scheduling algorithm, 101, 644
- shutdown* system call, 441, 462, 541
- sigaction* system call, 113, 117, 618
- SIGALRM, 66
- sigaltstack* system call, 113
- SIGCHLD, 116, 120, 130
- SIGCONT, 113, 116–117, 619
- SIGHUP, 123, 420, 429
- SIGINT, 69
- SIGIO, 229, 426, 451, 644

- SIGKILL, 28, 113, 117
- signal, 28, 84, 111–123, 644
 - checking for a pending, 55
 - comparison with other systems, 114
 - delivering, 117–118
 - driven I/O, 229, 234, 644
 - handler, 28, 111, 113, 644
 - masking, 113
 - posting, 113–117
 - priority, 28
 - restrictions on posting, 113
 - stack, 28, 113
 - trampoline code, 118, 644
- sigpause* system call, 91
- sigprocmask* system call, 113, 632
- SIGPROF, 66, 75
- sigreturn* system call, 114, 118, 644
- SIGSTOP, 28, 113, 131
- sigsuspend* system call, 113
- sigtramp()*, 118
- SIGTRAP, 130–131
- SIGTSTP, 133, 427
- SIGTTIN, 122–123, 427
- SIGTTOU, 116, 122–123, 424
- SIGURG, 451
- SIGVTALRM, 66
- SIGWINCH, 419
- silly-window syndrome, 547, 644
 - TCP handling of, 547–548
- single indirect block, 299, 627, 644
- SI_ORDER_FIRST, 600
- SI_ORDER_SECOND, 600
- Sixth Edition UNIX, 4–5
- size update, filestore, 359
- sleep()*, 86–87, 89–91, 102, 104, 113, 115, 190, 217, 457, 644, 649
 - implementation of, 86–87, 90–91
 - interruptible, 86, 115
 - operation of, 91
 - use of *sleep()*, 86–87, 90
- sleep queue, 85, 644
- sliding-window scheme, 528, 645
- slow-start algorithm, TCP, 550–554
- small computer system interface, 265–268, 277–279, 284–286
 - bus, 283, 644
 - disk, 277
- small-packet avoidance, 584, 645
 - TCP implementation of, 548–549
- SMP. See *symmetric multiprocessing*
- SMT. See *symmetric multithreading*
- snapshot, 349, 645
 - on a large filesystem, 353–354
 - creating a, 349–352
 - deadlock, 353–354
 - maintaining a, 352–353
 - performance, 355–356
 - user visible, 357–358
- socantrcvmore()*, 545
- sockaddr_dl*, 480
- sockaddr_in*, 564
- sockaddr_in6*, 564
- socket, 33, 36, 44, 215, 226, 437, 449, 473, 645
 - address, 453–454
 - address structure, 438, 645
 - connection queueing, 452, 455
 - data buffer locking, 460
 - data buffering, 451, 459, 461
 - data structures, 451–453
 - error handling, 457
 - options, 484
 - process group association with, 122, 451
 - shutdown, 463–464
 - state transitions during rendezvous, 456
 - state transitions during shutdown, 464
 - states, 453
 - types, 437, 449
 - using a, 438–441
- socket* system call, 8, 33–35, 44–45, 227, 438–439, 449, 455, 484, 488, 621
 - definition, 438
- socket-to-protocol interface, 484–488
- socketpair* system call, 487, 621
- SOCK_STREAM, 438
- soconnect()*, 457
- soft limit, 72, 645
- soft link, 314, 645, 647. See also *symbolic link*
- soft updates, 324–349
 - bitmap dependencies, 332
 - dependencies, 325–329
 - direct block dependencies, 334–335
 - directory dependencies, 338–341
 - fsck** dependencies, 347
 - fsync* dependencies, 344–345
 - indirect block dependencies, 336–338
 - inode dependencies, 332–334

- link count dependencies, 345–347
- overview, 324–325
- performance, 347–349
- reclamation dependencies, 342–344
- structures, 329–332
- truncation dependencies, 341–343
- softclock()*, 58–61, 66
- softdep_disk_io_initiation()*, 334
- softdep_disk_write_complete()*, 334
- softdep_update_inodeblock()*, 334
- software interrupt, 57, 645
- sohasoutofband()*, 545
- soisconnected()*, 457
- soisconnecting()*, 538
- soisdisconnected()*, 545
- SO_LINGER, 464
- solisten()*, 455
- sonewconn()*, 455, 540
- soreceive()*, 394, 460, 462–463, 469
- sorwakeup()*, 462
- sosend()*, 394, 458–460, 469, 546, 584
- source-quench processing, TCP, 552
- Spec 1170, 8
- special-device, 226
- special file, 35, 226, 645
- SPI. See *security-parameter index*
- spin mutex, 94–97, 645
- ssh**, 43, 413, 608
- stack, 645
 - growth on PC, 63
 - segment, 29, 61, 169, 645
 - zero filling of user, 63
- stackable filesystem, 253–259
 - 4.4BSD, 253–254
- stale data, 401
- stale translation, 184–185, 646
- standalone
 - device driver, 595, 646
 - I/O library, 595, 646
 - program, 595–596, 646
- standard error, 34, 646
- standard input, 34, 646
- standard output, 34, 646
- start kernel threads, 606–607
- start routine, 278, 280, 425, 430
- stat* structure, 227, 303, 487
- stat* system call, 40, 246, 253, 303, 310, 643
- statclock()*, 58–59, 71
- stateless protocol, 391, 646
- statfs* system call, 245
- statistics collection, 58–59, 71
- statistics, system, 58–59
- sticky bit, 209, 646
- stop character, 425
- storage-management algorithm, mbuf, 447–448
- strategy()*, 252, 277–278, 280
- strategy* vnode operator, 334
- stream I/O system, 6
- stream socket, 437, 646
- STREAMS, 235, 489
- structure, thread, 87–88
- structures, soft updates, 329–332
- su**, 304
- subnet, 516
- Sun Microsystems, 9, 42, 240, 242, 376, 386, 389, 393, 396, 399, 419
- superblock, 363, 647
- superuser, 66, 231, 647
- supplementary group array, 68
- swap
 - area, 140, 647
 - device, 140, 647
 - out, 82, 194–195
 - out, 4.4BSD, 195
 - pager, 154, 177–180
 - pager, 4.4BSD, 178
 - partitions, 178
 - space, 140, 177, 647
 - space management, 177–180
- swapon()*, 92
 - operation of, 195–196
- swapoff**, 179
- swapper*, 50, 195–196, 601, 606, 629, 647
- swapping, 30, 64, 139–140, 194–196, 647
 - in FreeBSD, reasons for, 194
- swi_net*, 475–476
- swi_sched()*, 105
- swp_pager_async_iodone()*, 179
- symbolic link, 314–315, 647
- symlink* system call, 350
- symlink* vnode operator, 295
- symmetric cryptography, 581, 647
- symmetric multiprocessing, 93, 105–107, 447, 455, 581, 597, 599–600, 645, 647
- symmetric multithreading, 105–107, 645, 647
- sync* system call, 242, 350

- syncache, 540
- syncer*, 50, 260, 329, 606
- synchronization, 93–99
 - network time, 65–66
- /sys/kern/sched_4bsd.c*, 99
- /sys/kern/sched_ule.c*, 99
- syscall()*, 53
- sysctl* system call, 190, 258, 483, 526, 612–614
 - implementation of, 612–614
- syslogd**, 608
- system activity, 53, 647
- system call, 22, 25–26, 51–52, 647
 - handling, 31, 53–56, 89
 - interrupted, 54–55
 - result handling, 54–55
 - return from, 55–56
- system calls
 - accept*, 440, 453, 455–457, 469, 540, 564
 - access*, 254
 - adjtime*, 66
 - aio_error*, 230–231
 - aio_read*, 230, 260
 - aio_return*, 231
 - aio_suspend*, 231
 - aio_waitcomplete*, 231
 - aio_write*, 230, 260
 - bind*, 519, 564
 - chdir*, 39, 620
 - chflags*, 303, 350
 - chmod*, 40, 350
 - chown*, 40, 350
 - chroot*, 39, 124–127, 640
 - close*, 33, 228, 231, 247, 253, 402–403, 416, 441, 463–464, 541
 - connect*, 439, 455, 457, 519–520, 522, 538, 564, 579, 619
 - dup*, 35, 41, 228–229, 621, 624
 - dup2*, 35, 229, 624
 - exec*, 34, 64, 67–69, 73, 79, 108, 119–120, 131, 146, 164, 166–168, 172, 174, 204, 209, 228, 609, 639, 641–643
 - exit*, 27, 108, 110, 167, 172
 - extattrctl*, 350
 - fchdir*, 125, 620
 - fchflags*, 303, 350
 - fchmod*, 40, 350
 - fchown*, 40, 350
 - fcntl*, 8, 228–230, 426, 621
 - fhopen*, 350
 - flock*, 388
 - fork*, 4, 27, 34, 41, 73, 79, 84–85, 90, 108, 110, 119–120, 130–131, 164–167, 174, 190, 204, 206, 209, 228, 618, 624, 635, 637, 639
 - fstat*, 40, 487
 - fsync*, 159, 242, 249, 260, 300–301, 327–328, 338, 344–345, 350, 362, 369, 376, 380, 402
 - ftruncate*, 350
 - futimes*, 350
 - getdirentries*, 309
 - getfsstat*, 245
 - getlogin*, 609
 - getpeername*, 441
 - getrusage*, 71
 - getsockname*, 441
 - getsockopt*, 441, 484, 488
 - gettimeofday*, 65–66
 - ioctl*, 35, 122, 227, 230, 415–418, 420, 428–429, 462, 479, 483, 487, 619
 - jail*, 126–127
 - jailkill*, 129
 - kill*, 113
 - killpg*, 122, 637
 - kse_create*, 83
 - ktrace*, 350
 - lchmod*, 350
 - lchown*, 350
 - link*, 40, 350
 - listen*, 439, 455, 540, 630
 - lseek*, 34, 227, 624
 - lstat*, 315
 - lutimes*, 350
 - mkdir*, 40, 46, 340, 350
 - mkfifo*, 350
 - mknod*, 350
 - mlock*, 159, 188, 207
 - mmap*, 30, 32, 64, 142, 158, 165, 170–171, 174, 204, 248, 632
 - mount*, 37, 222, 253, 256, 258, 394–395, 608
 - mprotect*, 171, 206
 - msgrcv*, 467–468
 - msgsnd*, 467
 - msync*, 159, 174, 176–177
 - munlock*, 159
 - munmap*, 158, 161, 165, 170, 205

- nfssvc*, 395–396
- open*, 33, 35, 41, 174, 222, 227, 253, 296–297, 306, 315, 350, 416, 423, 441, 465, 621
- pipe*, 33, 35, 227, 621
- poll*, 222, 234–236, 416, 636
- profil*, 75
- ptrace*, 93, 129–131
- quotactl*, 350
- read*, 31, 33, 36–37, 44, 131, 227, 236, 240, 248, 253, 260, 415, 426–428, 441, 458, 624, 634, 638, 647
- readv*, 36–37, 239, 628
- reboot*, 610, 614
- recv*, 36–37
- recvfrom*, 36–37, 458, 564
- recvmsg*, 36, 458, 463
- rename*, 40, 350, 391
- revoke*, 247, 350, 420, 429, 608
- rfork*, 83–84, 108
- rmdir*, 40, 340, 345, 350
- rtprio*, 86
- sbrk*, 63, 141–142, 165, 169, 626
- select*, 222, 234–238, 260, 452, 540, 636
- semctl*, 470
- semget*, 466, 470
- semop*, 466, 470
- send*, 36–37, 44, 451, 576
- sendfile*, 32
- sendmsg*, 36, 458, 484, 519
- sendto*, 36–37, 458, 484, 519, 564
- seteuid*, 69
- setlogin*, 609
- setpgid*, 119–120
- setpriority*, 637
- setsid*, 120
- setsockopt*, 441, 464, 484, 488, 517, 520, 549, 619
- settimeofday*, 65
- shmat*, 468
- shmdt*, 468, 470
- shmem*, 157, 468
- shmget*, 468
- shutdown*, 441, 462, 541
- sigaction*, 113, 117, 618
- sigaltstack*, 113
- sigpause*, 91
- sigprocmask*, 113, 632
- sigreturn*, 114, 118, 644
- sigsuspend*, 113
- socket*, 8, 33–35, 44–45, 227, 438–439, 449, 455, 484, 488, 621
- socketpair*, 487, 621
- stat*, 246, 253, 303, 310, 643
- statfs*, 245
- symlink*, 350
- sync*, 242, 350
- sysctl*, 190, 258, 483, 526, 612–614
- tcsetattr*, 623, 630, 652
- truncate*, 41, 346, 350
- undelete*, 258, 350
- unlink*, 40, 346, 350, 391
- unmount*, 253, 350
- utimes*, 303, 350
- vfork*, 84, 108, 119, 167, 209
- wait*, 27, 71, 79, 85, 90–91, 119, 131, 167, 172
- wait4*, 27, 110–111, 130
- write*, 26, 31, 33, 35–37, 44, 131, 227, 234, 236, 240, 260, 316, 350, 368–369, 397, 402–403, 415, 425, 441, 451, 458, 576, 624, 638, 647
- writev*, 36–37, 239, 628
- system crash, 40, 42, 223, 231, 233, 249, 284, 319, 324, 326, 328, 347, 357, 362–363, 380, 391–392, 396–399, 402–404, 408–410, 529, 531, 536, 599, 603, 608, 611–612, 620
- system debugging, 611–612
- system entry, 50–51
- system error, EAGAIN, 110, 229, 427, 460, 465–466, 634
- system error, EINTR, 55, 87, 110
- system performance, 54, 57, 61, 63, 65, 80, 104, 460
- system security, 123–129
- system shutdown, 610–611
- system startup, 593–595
 - initial state, 597
 - scripts, 607–608
- system statistics, 58–59
- System V, xix, xix, xxi, 11–12, 84, 604
 - message queue, 435, 467–468
 - poll interface, 235
 - range lock, 231
 - semaphores, 464–466, 605
 - shared memory, 157, 176, 468
 - terminal driver, 416

T

- T-shirt, daemon, xxiv
- table, forward-mapped page, 197, 624
- tags, 447, 648
- tasklist*, 329, 335, 341–343
- TCB. See *thread control block*
- TCP. See *transmission control protocol*
- tcp_attach()*, 538
- tcpcb* structure, 518, 538
- tcp_close()*, 541
- tcp_connect()*, 538
- tcp_ctloutput()*, 549
- tcp_delack()*, 545
- tcp_hc_get()*, 539
- tcp_hc_purge()*, 539
- tcp_input()*, 534, 543, 545, 556
 - operation of, 542–545
- tcp_output()*, 534, 538, 543, 545–547, 549–550
 - operation of, 547
- tcp_slowtimo()*, 535
- tcp_usr_send()*, 534, 546
- tcp_usr_shutdown()*, 541
- tcsetattr* system call, 623, 630, 652
- tcsetpgrp()*, 122
- TDF_NEEDRESCHED, 105, 117
- telldir()*, 309
- telnet**, 43, 413
- TENEX operating system, 7
- Tenth Edition UNIX, 5
- terminal, 43, 413, 648
 - buffering, 421–422
 - input, 426–428
 - multiplexer, 215
 - operations, 423–430
 - output, 426
- terminal driver, 415–416, 649
 - bottom half of, 415
 - close()*, 429–430
 - data queues, 419, 421–422, 424, 427–428
 - functions, 416–418
 - hardware state, 419
 - ioctl()*, 416–418, 428–429
 - modem control, 422–423
 - modem transitions, 429
 - modes, 414–415, 419, 426
 - open()*, 423
 - software state, 419
 - special characters, 414, 419
 - System V, 416
 - top half of, 415
 - user interface, 7, 416–418
 - window size, 419, 423
- terminal process group, 122, 420, 426–427, 429
- termios structure, 416, 648
- text segment, 29, 61, 63, 648. See also *shared text segment*
- Thompson, Ken, 3–4, 7, 22
- thrashing, 82, 648
- thread, 82, 156, 648
 - control block, 52, 83, 88–90, 648
 - mutex, network, 494
 - preemption, 105
 - priority, 85–86, 90
 - priority, calculation of, 92, 101–102
 - priority, while sleeping, 86
 - queues, 85
 - scheduling, 93, 100–108
 - state, 89
 - state, change of, 92
 - structure, 87–88
- thread_exit()*, 110
- 3BSD, 6
 - filesystem, 363, 366, 370–371
- tick, 57, 648
- time, 57–59, 65–66
 - of day, 50
 - of-day register, 65
 - interval, 66
 - process virtual, 66
 - quantum, 100, 648
 - representation, 65
 - slice, 81, 100, 648
 - stable identifier, 391, 648
 - synchronization, network, 65–66
 - wall clock, 65–66
- time zone handling, 9
- timeout()*, 59–61
- timer
 - 2MSL, 536–537, 649
 - backoff, 536, 648
 - network, 59, 477
 - profiling, 58, 66
 - real-time, 59, 66
 - resolution of, 65
 - routines, TCP, 535

- virtual-time, 58, 66
 - watchdog, 59
 - timestamps option, TCP, 530, 538
 - TLB. See *translation lookaside buffer*
 - /tmp**, 158
 - top half of, 51, 649
 - device driver, 217
 - kernel, 51–52
 - terminal driver, 415
 - TOPS-20 operating system, 7
 - Toy Story*, xxiv
 - trace trap, 130, 649
 - traced process, 116, 130
 - track cache, 375–377, 649
 - translation lookaside buffer, 197–198, 204–208, 649
 - transmission control protocol, xx, 3, 6, 45, 127–128, 259, 388–389, 395, 399–400, 410, 476, 499, 502, 508, 514–515, 517–519, 526–557, 584–586, 626, 638, 648–649
 - algorithm, 534–541
 - congestion control, 550–554
 - connection setup, 529–530, 538–541
 - connection shutdown, 530, 541
 - connection states, 529–531
 - data buffering, 553
 - delayed acknowledgments in, 545, 549
 - estimation of round-trip time, 537–538
 - fast retransmission, 554–556
 - features of, 527
 - flow control in, 528
 - handling of silly-window syndrome, 547–548
 - handling of urgent data, 545
 - header prediction, 543, 626
 - host cache, 538
 - host cache metrics, 539
 - implementation of small packet avoidance, 548–549
 - implementation, use of 4BSD, 8
 - input processing, 542–545
 - maximum-segment-size option, 530, 538
 - options, 528
 - output processing, 546–556
 - packet header, 529
 - receive window, 542
 - retransmission handling, 550
 - send policy, 534–535, 546–556
 - sequence numbers, 528
 - sequence variables, 531–534
 - slow-start algorithm, 550–554
 - source-quench processing, 552
 - state diagram, 532
 - syncache, 540–541
 - timer routines, 535
 - timers, 535–537
 - timestamps option, 530, 538
 - window-scale option, 530
 - window updates, 549
 - transport layer, 474, 649
 - transport mode, 570, 649
 - security association, 571
 - trap()*, 53
 - trap handling, 50, 53–54, 56–57, 89
 - trap type code, 53
 - triple indirect block, 299, 627, 649
 - truncate* system call, 41, 346, 350
 - addition of, 41
 - truncate* vnode operator, 359
 - truncation dependencies, soft updates, 341–343
 - ttioctl()*, 428–429
 - ttread()*, 427
 - ttselect()*, 222
 - ttstart()*, 425, 430
 - ttwakeup()*, 426
 - ttwrite()*, 424–425, 427
 - tty driver*, 415
 - tty driver. See *terminal driver*
 - tty* structure, 418–419
 - ttyclose()*, 430
 - ttyinput()*, 426–427
 - ttylclose()*, 429
 - ttymodem()*, 429
 - ttyoutput()*, 425
 - ttypoll()*, 415
 - Tunis operating system, 7, 22
 - tunnel mode, 571, 649
 - security association, 571
 - tunneling, 474
 - 2MSL timer, 536–537, 649. See also *maximum segment lifetime*
 - type-ahead, 414, 650
- ## U
- UDP. See *user datagram protocol*
 - udp_append()*, 521
 - udp_attach()*, 519

- udp_bind()*, 519
 - udp_detach()*, 522
 - udp_input()*, 521
 - udp_output()*, 520
 - udp_send()*, 520
 - UFS, 311, 324, 365
 - UFS1, 299–300, 303–305, 311, 363–366, 378
 - UFS2, 299–300, 303–305, 311, 345–346, 364–366, 378
 - ufs_bmap()*, 368, 377
 - UID. See *user identifier*
 - uio* structure, 220–221, 239–240, 359, 424, 427–428, 650
 - uiomove()*, 222, 240, 425
 - implementation of, 239–240
 - ULE scheduler, 99–100, 105–108
 - umapfs** filesystem, 256, 400
 - uma_zalloc()*, 150
 - uma_zcreate()*, 150
 - uma_zfree()*, 150
 - uma_zone_set_max()*, 150
 - undelete* system call, 258, 350
 - union** filesystem, 256–258
 - Universal Coordinated Time, 65, 74
 - universal serial bus, 265–268, 277, 292
 - University of California at Berkeley, 6
 - UNIX
 - 32V, 5–6
 - history of, 3–7
 - Programmer’s Manual, 4
 - Support Group, 5–6
 - System III, 5–6, 8
 - System Laboratory, 6, 11–13
 - System V, 5–8
 - System V, Release 3, 6
 - United Filesystem, 385
 - unlink* system call, 40, 346, 350, 391
 - unlock* vnode operator, 296–297
 - unmount**, 345
 - unmount* system call, 253, 350
 - unputc()*, 422
 - update dependency, 326, 650
 - update* vnode operator, 334–335, 338, 341–342, 358
 - updatepri()*, 103
 - ureadc()*, 427
 - urgent data, 508, 650
 - TCP handling of, 545
 - transmission, styles of, 460
 - urgent delivery*, 508
 - USB. See *universal serial bus*
 - use of descriptor, 33–34
 - USENET, 9, 378
 - user datagram protocol, 127–128, 388–389, 391, 395, 398–400, 410, 502, 514–515, 517–522, 527, 535, 546, 556–557, 584–585, 638, 650
 - control operations, 521–522
 - initialization, 519–520
 - input, 521
 - output, 520–521
 - user identifier, 66–69, 73, 127–128, 255–256, 388, 400–401, 622, 627, 639, 642–643, 647, 650
 - use in file-access validation, 67
 - user-level system initialization, 607–609
 - user mode, 79, 140, 650
 - user-request routine, 650
 - user request routines, 478, 484–487
 - operations, 484–487
 - user structure, 52, 80, 650
 - USL. See *UNIX System Laboratory*
 - /usr/sbin/config**, 292
 - UTC. See *Universal Coordinated Time*
 - utimes* system call, 303, 350
- ## V
- V Kernel operating system, 22
 - valloc* filesystem operator, 358
 - /var/quotas**, 316
 - /var/run/lock**, 396
 - VAX, 5–7
 - memory management hardware, 30
 - vfork* system call, 84, 108, 119, 167, 209
 - implementation issues, 167
 - operation of, 167
 - see also *process creation*
 - vfree* filesystem operator, 358
 - vfs.usermount**, 258
 - vget* filesystem operator, 358
 - vgone()*, 247
 - virtual-address aliasing, 198, 651
 - virtual address space, 136, 651
 - layout of user, 61–65
 - process, 151
 - virtual-filesystem interface, 240–245
 - virtual memory, 6, 651
 - 4.4BSD, 31

- for a shared-memory multiprocessor, 30
 - advantages of, 140
 - cache coherency, 175
 - change protection, 171
 - change size, 169
 - data structures, 142–144
 - duplication, process, 166–167
 - hardware requirements for, 140–141
 - implementation portability, 196–209
 - initialization, 201–204, 208
 - interface, 4.2BSD, 7
 - layout, 141–142
 - machine dependencies, 196–209
 - manipulation of, 169–171
 - map allocation, 204–206
 - map protection, 206–207
 - maps, 145–146
 - object, 153–156, 651
 - overview, 141–144
 - resources, process, 151–156
 - semaphores, 156–157
 - usage calculation of, 165–166, 169
 - virtual private network, 571
 - virtual-time timer, 58, 66
 - vm_daemon()*, 50, 195
 - vm_fault()*, 71, 180–182, 199, 207
 - vm_forkproc()*, 110
 - vm_map* structure, 143, 145–146, 200
 - vm_map_entry* structure, 143, 145–146, 149, 151–154, 159, 161, 164, 166, 168–172, 180, 204, 206, 210
 - vm_object* structure, 143–144, 241
 - vm_page* structure, 144, 153, 155, 173–176, 187, 202, 205, 208
 - vm_page_alloc()*, 189
 - vm_page_io_finish()*, 179
 - vm_pageout()*, 190
 - vm_pageout_scan()*, 190, 192–193
 - vm_page_startup()*, 202
 - vm_page_test_dirty()*, 207–208
 - VMS operating system, 7–8, 188
 - page replacement in the, 188
 - vmSPACE* structure, 143, 151–152, 164, 167–168
 - vmSPACE_exec()*, 209
 - vmSPACE_fork()*, 209
 - vmSPACE_free()*, 209
 - vnLru*, 50, 606
 - vnnode*, 37, 226, 240, 451, 651
 - cache, 155
 - description of, 241–244
 - operations, 242–244
 - vnnode* operator
 - access*, 296
 - advlock*, 296
 - blkatoff*, 359
 - close*, 296
 - create*, 295, 297
 - fsync*, 359
 - getattr*, 296
 - inactive*, 246–247, 296, 307
 - ioctl*, 296
 - link*, 295
 - lock*, 296
 - lookup*, 244, 296
 - mknod*, 295
 - mknod*, 295
 - mmap*, 296
 - open*, 296
 - poll*, 296
 - read*, 358–359
 - readdir*, 296
 - readlink*, 296
 - reclaim*, 246–247, 296, 307
 - remove*, 295
 - rename*, 295
 - rmdir*, 296
 - setattr*, 296
 - strategy*, 334
 - symlink*, 295
 - truncate*, 359
 - unlock*, 296–297
 - update*, 334–335, 338, 341–342, 358
 - write*, 359
 - vnnode* pager, 154, 174–175
 - vnnode_pager_setsize()*, 174
 - voluntary context switching, 89–93
 - vop_access_args* structure, 255
- ## W
- wait channel, 88, 90–92, 98, 651
 - wait* system call, 27, 71, 79, 85, 90–91, 119, 131, 167, 172, 651
 - wait4* system call, 27, 110–111, 130
 - operation of, 111

- `wakeup()`, 91–93, 103, 105, 107, 179
 - implementation of, 92–93
 - operation of, 92
 - `wakeup_one()`, 92
 - wall clock time, 65–66
 - watchdog timer, 59
 - whiteout, filename, 257
 - wildcard route, 496, 521, 651
 - window probe, 536, 651
 - window-scale option, TCP, 530
 - window size, 419, 423
 - window system, 122, 419. See also *X Window System*
 - Windows operating system, xx
 - wine, xxvii
 - wired page, 146, 176, 199–200, 202, 207, 651
 - definition of, 146
 - list, 188
 - word-erase character, 414, 652
 - working set, 139, 652
 - worklist* structure, 329, 332
 - workstation, 135
 - write-caching lease, 405–406, 652
 - `write` system call, 26, 31, 33, 35–37, 44, 131, 227, 234, 236, 240, 260, 316, 350, 368–369, 397, 402–403, 415, 425, 441, 451, 458, 576, 624, 638, 647
 - `write` vnode operator, 359
 - `write_slack`, 404, 406, 408–409
 - `writew` system call, 36–37, 239, 628
- ## X
- X/OPEN, xix, 6, 8
 - X Window System, 419, 548
 - X.25, 492
 - XDR. See *external data representation*
 - Xerox NS protocols, 513
 - xform-switch* structure, 580–581
 - XINU operating system, 7
 - `xpt_action()`, 278–279
 - `xpt_done()`, 279
 - `xpt_schedule()`, 278
 - XPT_SCSI_IO, 278
 - xterm**, 43, 413
- ## Z
- zalloc, 149–150
 - `zalloc()`, 32
 - zero filling of user stack, 63
 - `zfree()`, 32
 - zombie process, 85, 111, 652
 - zone allocator, 149–150
 - zone, red, 32, 147, 640