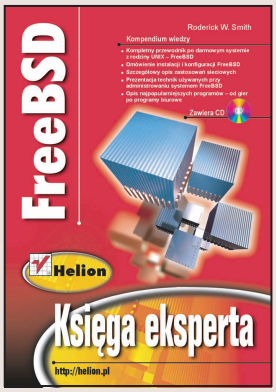
FreeBSD. Księga eksperta



Autor: Roderick W. Smith
Tłumaczenie: Marcin Jędrysiak
ISBN: 83-7361-158-4
Tytuł oryginału: FreeBSD: The Complete Reference
Format: B5, stron: 824

FreeBSD to zaawansowany system operacyjny dla komputerów opartych o architekturę x86, DEC Alpha, IA-64, PC-98 i UltraSPARC. Ma długą historię, wywodzi się bowiem od BSD, wersji UNIX-a, stworzonej na Uniwersytecie Kalifornijskim w Berkeley. Jest on systemem darmowym, opartym na bardzo liberalnej licencji.

FreeBSD może być uruchomiony na komputerze domowym i służyć do surfowania po internecie, pisania listów i grania. Użytkownik może także zainstalować FreeBSD w celu uruchamiania edytorów tekstu i arkuszy kalkulacyjnych. FreeBSD jest często stosowany przez naukowców i inżynierów do analizy złożonych danych oraz tworzenia nowych narzędzi w wielu językach programowania. Oczywiście system może być wykorzystany także na serwerach sieciowych, które poprzez Apache, Sambę i Sendmail dostarczają informacji użytkownikom w sieci lokalnej i internecie. FreeBSD może być zastosowany nawet na komputerze służącym jako router i firewall, chroniący sieć przed atakami z zewnątrz. Ta książka omawia wszystkie te potencjalne sposoby użycia FreeBSD.

W książce opisano m.in. zagadnienia:

- Instalacja systemu FreeBSD
- Współpraca FreeBSD z innymi systemami operacyjnymi
- Podstawowe narzędzia dostarczane w FreeBSD
- Zarządzanie partycjami i plikami
- Zarządzanie kontami użytkowników
- Instalowanie oprogramowania i konfiguracja jądra systemu
- System X Window
- Konfiguracja sieci
- Zapora sieciowa w FreeBSD
- Serwery: FTP, NFS, Samba, SMTP, POP3/IMAP, HTTP
- Zdalny dostęp: Telnet, SSH, X, VNC
- Omówienie najpopularniejszych programów
- Multimedia i gry
- Kompilowanie oprogramowania
- Zabezpieczanie systemu

Jeżeli potrzebujesz wydajnego i bardzo bezpiecznego systemu operacyjnego, warto zastanowić się nad FreeBSD. Ta książka nauczy Cię wszystkiego, co potrzebne, by używać i administrować tym systemem.

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

Spis treści

O Autorze 19

Wprowadzenie 21

Część I Instalacja FreeBSD 27

Rozdział 1. Wymagania FreeBSD 29

Dlaczego FreeBSD? 29
FreeBSD jako system operacyjny dla stacji roboczej 29
FreeBSD jako system operacyjny dla serwera 31
FreeBSD a inne systemy nie-uniksowe 32
FreeBSD a inne systemy uniksowe 34
Środowisko pracy 38
Historia FreeBSD 38
Interakcja z innymi komputerami w sieci 39
Interakcja z innymi systemami na tym samym komputerze 41
Środowisko sprzętowe 43
Typy procesorów i ich prędkość 43
Wymagania wobec pamięci 44
Dyski twarde 45
Karta graficzna 48
Urządzenia sieciowe 49
Drukarki 51
Inne urządzenia 52
Podsumowanie 53

Rozdział 2. Instalacja systemu 55

Uzyskanie FreeBSD 55
Kwestie licencjonowania FreeBSD 55
Źródła FreeBSD 57
Przygotowanie nośników instalacyjnych 58
Przygotowanie komputera do uruchomienia FreeBSD 59
Sprawdzanie zgodności urządzeń 60
Przygotowanie przestrzeni dyskowej 62
Rozpoczęcie instalacji 66
Uruchomienie programu instalacyjnego 66
Partycjonowanie dysku 67
Wybór oprogramowania do instalacji 72
Wybór nośnika instalacji 73
Instalacja FreeBSD 74

Konfiguracja po zakończeniu instalacji....74
Konfiguracja sieci....74
Ustawienia konsoli systemowej....76
Strefa czasowa....77
Zgodność binarna z Linuksem....77
Konfiguracja myszy....78
Konfiguracja XFree86....79
Konfiguracja pulpitu X....82
Konfiguracja pakietów FreeBSD....83
Konfiguracja kont użytkowników....83
Zakończenie instalacji....88
Podsumowanie....88

Rozdział 3. Przegląd systemu....89

Uruchamianie komputera....89
Wybór systemu startowego....89
Interpretacja komunikatów startowych....90
Skrypty startowe....92
Logowanie się do systemu....93
Logowanie się w trybie tekstowym i graficznym....94
Konto użytkownika a konto root....95
Podsumowanie poleceń trybu tekstowego....97
Przedstawienie powłok....97
Uruchamianie programów w trybie tekstowym....99
Polecenia do manipulacji plikami....101
Przedstawienie graficznego interfejsu użytkownika FreeBSD....105
Uruchamianie X....105
Manipulacja plikami....107
Uruchamianie programów w środowisku graficznym....108
Konfiguracja środowiska graficznego....110
Wylogowywanie się i zamykanie systemu....111
Wylogowywanie się w trybie tekstowym i graficznym....111
Zamykanie systemu w trybie tekstowym i graficznym....112
Kiedy należy wyłączać komputer?....114
Podsumowanie....114

Rozdział 4. Współpraca z innymi systemami....117

Konfiguracja procesu startowego....117
Działanie programu startowego....118
Użycie programu startowego FreeBSD....119
Ustawianie opcji jądra....121
Użycie alternatywnego programu startowego....122
Zarządzanie partycjami FreeBSD i x86....123
Przegląd systemów partycjonowania....124
Ochrona partycji FreeBSD przed innymi systemami operacyjnymi....124
Uzyskanie dostępu do obcych partycji we FreeBSD....126
Zrozumienie systemów plików....128
Dwa znaczenia określenia „system plików”....128
Podstawy systemu plików FreeBSD....128
Inne systemy plików....129

Dostęp do obcych systemów plików z poziomu FreeBSD....133
Montowanie obcych systemów plików....133
Ograniczenia w użyciu obcych systemów plików....135
Dostęp do systemu plików FreeBSD z poziomu innych systemów operacyjnych....137
Obsługa FFS w innych systemach operacyjnych....137
Uwagi dotyczące użycia partycji FreeBSD w innych systemach operacyjnych....138
Uruchamianie obcych programów we FreeBSD....138
Uruchamianie programów linuksowych....139
Użycie WINE do uruchamiania programów Windows....141
Użycie emulatora....144
Podsumowanie....145

Część II Podstawowa administracja systemem....147

Rozdział 5. Narzędzia administracyjne....149
Katalog /etc i jego zawartość....149
Pliki startowe systemu....150
Pliki konfiguracyjne programów serwerowych....151
Inne pliki konfiguracyjne....153
Przegląd edytorów tekstu we FreeBSD....154
Edytory tekstu pracujące w trybie tekstowym....154
Edytory tekstowe z graficznym interfejsem użytkownika....155
Przykład — edycja pliku w vi....156
Przykład — edycja pliku w gEdit....160
Przedstawienie najważniejszych poleceń do administracji systemem....163
Polecenia dostarczające informacji o systemie....164
Polecenia do manipulacji systemem....169
Przekierowanie....171
Potoki....173
Narzędzia administracyjne z graficznym interfejsem użytkownika....174
Podsumowanie....175

Rozdział 6. Uruchamianie systemu i procesy sterujące....177
Przedstawienie procesów sterujących jądra....177
Ładowanie jądra....178
Kontrola jądra nad kolejnymi etapami procesu startowego....178
Rola jądra w obsłudze systemu....179
Modyfikacja skryptów startowych....180
Edycja skryptów startowych systemu....181
Tworzenie nowych lokalnych skryptów startowych....182
Wewnątrz procesu logowania....183
Metody logowania użytkowników....183
Uwierzytelnianie użytkowników poprzez PAM....186
Modyfikacja domyślnych ustawień logowania....188
Pliki z ustawieniami użytkownika....189
Użycie narzędzia cron do uruchamiania programów o określonych porach....192
Użycie narzędzia at....192
Przedstawienie narzędzia cron....193

Tworzenie systemowych zadań cron....194
Tworzenie zadań cron użytkownika....196
Podsumowanie....196

Rozdział 7. Zarządzanie partycjami.... 197

Partycje — kontenery dla plików z danymi....197
Przyczyny partycjonowania dysku....197
Tworzenie partycji....199
Montowanie partycji....206
Odłączanie partycji....210
Użycie tradycyjnych schematów partycjonowania w Uniksie....211
Które katalogi mogą być partycjami?....211
Typowe schematy partycjonowania....212
Montowanie partycji innego typu....213
Zrozumienie FHS....213
Rola FHS....214
Przegląd katalogów FHS....215
Zgodność FreeBSD ze standardem FHS....217
Podsumowanie....217

Rozdział 8. Zarządzanie plikami.... 219

Prawa własności i uprawnienia....219
Powiązania między kontami i własnością plików....219
Użycie uprawnień do kontroli dostępu do plików....221
Uprawnienia katalogów....224
Projektowanie schematu uprawnień systemowych....224
Polecenia do manipulacji plikami....225
Zmiana praw własności....226
Zmiana uprawnień....227
Kopiowanie i przenoszenie plików....229
Usuwanie plików....232
Tworzenie i użycie dowiązań....234
Wyszukiwanie plików....235
Archiwizacja danych....240
Nośniki archiwizacji....241
Przegląd narzędzi do archiwizacji danych....241
Użycie narzędzia tar....242
Przywracanie danych....245
Podsumowanie....246

Rozdział 9. Konfiguracja drukarek.... 247

Zrozumienie modelu drukowania FreeBSD....247
Koncepcja kolejek wydruków....247
Przesyłanie zadań wydruku do kolejki....249
Rola programu Line Printer Daemon....249
PostScript jako standardowy język drukowania....250
Tworzenie kolejki wydruków....251
Użycie plików urządzeń....252
Tworzenie katalogu buforowania....254
Format pliku /etc/printcap....255
Użycie prostego filtru....257

Konfiguracja programu Ghostscript 259
Użycie programu Ghostscript jako translatora 259
Konfiguracja inteligentnego filtru 261
Sterowanie drukarką 263
Przesyłanie zadań wydruku 263
Kontrola zadań drukowania 265
Usuwanie zadań wydruku 266
Zmiana kolejności zadań wydruku 267
Zmiana dostępności kolejki 267
Kontrolowanie dostępu do zdalnej drukarki 268
Podsumowanie 269

Rozdział 10. Zarządzanie kontami użytkowników ... 271

Znaczenie kont we FreeBSD 271
Korzyści płynące z użycia kont 271
Konto jako narzędzie zabezpieczeń 272
Konta i grupy 273
Nazwy użytkowników, grupy oraz identyfikatory UID i GID 274
Struktury danych konta i grupy 275
Tworzenie kont użytkowników 276
Zasady tworzenia nazw użytkowników 276
Użycie programu sysinstall do tworzenia kont 277
Użycie narzędzia adduser 279
Modyfikacja kont użytkowników 281
Przesłanianie haseł 282
Użycie polecenia passwd 282
Użycie polecenia chpass 283
Edycja pliku /etc/master.passwd 285
Zmiana identyfikatora użytkownika (UID) 286
Usuwanie kont użytkowników 287
Użycie narzędzia rmuser 287
Usuwanie użytkowników z pliku /etc/master.passwd 288
Usuwanie pozostałych plików użytkownika 289
Użycie grup 289
Użycie programu sysinstall do tworzenia grup 290
Edycja pliku /etc/group 290
Zapewnienie bezpieczeństwa kont 291
Wybór strategii grup 291
Ustawianie domyślnych uprawnień pliku 293
Podsumowanie 294

Rozdział 11. Instalacja oprogramowania ... 295

Formy pakietów oprogramowania 295
Kod źródłowy a kod binarny 296
Metody dystrybucji 297
Źródła oprogramowania FreeBSD 298
Źródła standardowego oprogramowania FreeBSD 298
Źródła oprogramowania uniksowego 299
Źródła oprogramowania komercyjnego 300

Instalacja pakietów 300
Użycie narzędzia sysinstall 301
Narzędzia do obsługi pakietów 303
Zbieranie informacji o pakietach 305
Użycie portów FreeBSD 306
Instalowanie i aktualizacja systemu portów 307
Kompilowanie i instalacja portu 309
Usuwanie i aktualizacja oprogramowania 311
Usuwanie oprogramowania 311
Aktualizacja oprogramowania 312
Weryfikacja autentyczności oprogramowania 314
Podsumowanie 315

Rozdział 12. Konfiguracja jądra 317

Omówienie funkcji jądra 317
Rola jądra we FreeBSD 317
Instalowanie i modyfikowanie jądra 318
Opcje startowe jądra 319
Ładowanie modułów jądra 320
Kompilowanie jądra 321
Uzyskanie kodu źródłowego jądra 321
Konfigurowanie jądra 322
Wykonanie kompilacji 325
Użycie nowego jądra 325
Aktualizowanie jądra 326
Kiedy należy dokonać aktualizacji jądra? 326
Instalowanie nowego jądra 327
Kompilowanie i użycie nowego jądra 327
Podsumowanie 328

Rozdział 13. X Window System 329

Zrozumienie X Window System 329
Model GUI 330
Sieciowa natura X 334
Zmiana konfiguracji X 335
Ogólne wskazówki dotyczące konfiguracji 336
Struktura pliku XF86Config 336
Ustawienia ekranu 338
Zmiana karty graficznej 339
Zmiana ustawień monitora 340
Zmiana konfiguracji myszy 341
Dodawanie i usuwanie czcionek 342
Narzędzia do zmiany konfiguracji X 345
Opcje X na poziomie użytkownika 346
Zmiana menedżera okien lub środowiska biurka 346
Zmiana prędkości śledzenia myszy 347
Zmiana prędkości powtarzania klawiszy 348
Uruchamianie wielu sesji X 349
Podsumowanie 350

Część III Konfiguracja sieci 351

Rozdział 14. Podstawowa konfiguracja sieci 353

Cechy sieci lokalnych i internetowych 353
Najważniejsze urządzenia sieciowe 354
Cechy TCP/IP 355
TCP/IP a inne porty sieciowe 356
Wprowadzenie do funkcji trasowania 357
Adresy IP, nazwy hostów i domeny 359
Użycie statycznego adresu IP 360
Uzyskanie adresu IP 360
Konfigurowanie z użyciem narzędzia sysinstall 361
Ręczne konfigurowanie 363
Użycie DHCP 367
Testowanie konfiguracji 368
Prosty test pingowania 368
Testowanie funkcji rozwiązywania nazw 370
Diagnozowanie problemów z trasowaniem 370
Testowanie zaawansowanych protokołów 371
Podsumowanie 372

Rozdział 15. Połączenia modemowe 373

Wybór dostawcy usług internetowych 373
Sprawdzenie działania modemu 374
Użycie programu dostępowego z graficznym interfejsem użytkownika 375
Wybór programu dostępowego 376
Utworzenie konfiguracji połączenia 376
Inicjowanie połączenia 380
Użycie programów dostępowych w trybie tekstowym 381
Edycja plików pomocniczych 381
Konfigurowanie obsługi PPP w jądrze 381
Użycie zdefiniowanej konfiguracji 385
Użycie PPPoE i łącza DSL 385
Czym jest PPPoE? 386
Pakiety PPPoE we FreeBSD 386
Konfigurowanie PPPoE 386
Użycie PPPoE 387
Podsumowanie 388

Rozdział 16. Koncepcje serwerów i klientów sieciowych 389

Zrozumienie roli klientów i serwerów 389
Zrozumienie portów i ich zastosowania 390
Rola numerów portów 391
Standardowe przypisania numerów portów TCP/IP 392
Porty TCP, UDP i inne typy portów 392
Śledzenie połączenia sieciowego 393
Inicjacja połączenia 394
Rola trasowania w sieci 394
Odpowiedź serwera 395
Firewalle i NAT 396

Metody uruchamiania serwerów 397
Ręczne uruchamianie serwerów 397
Użycie bezpośrednich skryptów startowych 397
Użycie superserwera 398
Użycie narzędzia netstat do diagnostyki sieciowej 400
Podstawy użycia narzędzia netstat 401
Uzyskanie informacji o interfejsach sieciowych 402
Uzyskanie statystyk sieciowych 402
Użycie narzędzia netstat w praktyce 402
Podsumowanie 403

Rozdział 17. Konfigurowanie firewalla 405

Zrozumienie zagrożenia 405
Ataki z zewnątrz 406
Złośliwi użytkownicy wewnątrz sieci 407
Nieautoryzowane programy 408
Typy firewalli 409
Firewalle filtrujące pakiety 409
Routery NAT 409
Serwery proxy 411
Narzędzia FreeBSD do obsługi firewalla 412
Narzędzia do filtrowania pakietów — ipfw i IP Filter 412
Typowe serwery proxy 413
Tworzenie skryptu firewalla 414
Niezbędne opcje jądra 414
Domyślna zasada 415
Wybór akceptowanych i odrzucanych typów ruchu 416
Projektowanie właściwych reguł 418
Automatyzacja skryptu firewalla 422
Konfigurowanie NAT 423
Podsumowanie 424

Część IV Serwery 425

Rozdział 18. Serwery plików 427

Typy serwerów plików 427
Serwery transmisji plików a serwery współdzielenia plików 427
Dodatkowe funkcje serwera plików 428
Typowe serwery plików i ich wykorzystanie 429
Przygotowanie serwera FTP 431
Typowe programy serwerowe FTP 431
Uruchamianie serwera FTP 432
Przygotowanie serwera NFS 433
Definiowanie eksportowanych systemów plików 433
Kwestie zabezpieczeń 435
Uruchamianie serwera NFS 436
Przygotowanie Samby 436
Konfigurowanie ustawień globalnych 437
Dopasowanie haseł 439

Definiowanie udziałów plików 440
Uruchomienie Samby 442
Podsumowanie 442

Rozdział 19. Serwery pocztowe 443

Typy serwerów pocztowych 443
Serwery typu push i pull 443
Przykłady serwerów push 446
Przykłady serwerów pull 446
Przygotowanie serwera SMTP 448
Kwestie konfiguracji domeny 448
Pliki konfiguracyjne serwera sendmail 448
Zmiana domyślnej konfiguracji 449
Blokowanie spamu 451
Przygotowanie serwera POP lub IMAP 457
Przygotowanie programu Fetchmail 458
Plik konfiguracyjny Fetchmail 459
Uruchomienie programu Fetchmail poprzez zadanie cron 461
Uruchomienie programu Fetchmail jako demona 462
Podsumowanie 462

Rozdział 20. Serwery WWW 463

Kiedy warto uruchomić serwer WWW? 463
Serwery WWW dla systemu FreeBSD 464
Podstawowa konfiguracja serwera Apache 465
Uruchamianie serwera Apache 466
Plik konfiguracyjny serwera Apache 467
Konfigurowanie podstawowych funkcji 467
Ustawianie opcji katalogu serwera 469
Użycie modułów Apache 471
Tworzenie stron internetowych 472
Podstawowa struktura języka HTML 473
Narzędzia do tworzenia stron internetowych 476
Projektowanie przenośnych stron internetowych 477
Podsumowanie 479

Rozdział 21. Serwery logowania 481

Typy serwerów logowania 481
Konfigurowanie serwera Telnet 483
Zrozumienie działania Telnetu 483
Uruchomienie serwera Telnet 484
Ograniczanie dostępu do serwera Telnet 484
Konfigurowanie serwera SSH 485
Przedstawienie zabezpieczeń protokołu SSH 486
Dostępne programy serwerowe SSH 487
Uruchomienie serwera SSH 487
Konfigurowanie opcji SSH 488
Konfigurowanie zdalnego dostępu X 489
Zrozumienie powiązania między klientem i serwerem X 490
Użycie X poprzez tryb tekstowy 491
Użycie protokołu XDMCP 495

Przygotowanie serwera XDMCP....495
Uruchomienie serwera VNC....500
Zrozumienie powiązania między klientem i serwerem VNC....500
Instalowanie VNC....501
Konfigurowanie serwera VNC....502
Nawiązanie połączenia VNC....504
Użycie VNC poprzez inetd....505
Podsumowanie....507

Rozdział 22. Inne typy serwerów....509
Serwery DHCP....509
Kiedy należy użyć serwera DHCP?....510
Instalowanie serwera DHCP....511
Podstawowa konfiguracja DHCP....512
Przydzielenie statycznych adresów IP....514
Serwery DNS....515
Kiedy należy użyć serwera DNS?....515
Podstawowa konfiguracja serwera....517
Konfigurowanie strefy....519
Serwery czasu....522
Omówienie protokołu NTP....522
Wyszukiwanie macierzystych serwerów czasu....523
Konfigurowanie NTP....524
Serwery czcionek....527
Obsługa czcionek w X Window System....527
Konfigurowanie serwera czcionek....528
Użycie serwera czcionek....530
Podsumowanie....531

Część V Najpopularniejsze programy....533

Rozdział 23. Środowiska pulpitu....535
Rola środowiska pulpitu....535
Przegląd dostępnych środowisk pulpitu....537
KDE....538
GNOME....540
XFce....542
Tworzenie własnego środowiska pulpitu....543
Korzystanie z GNOME....545
Uruchamianie GNOME....545
Operacje na plikach....546
Konfigurowanie pulpitu GNOME....546
Podsumowanie....554

Rozdział 24. Narzędzia sieciowe....555
Programy pocztowe....555
Przedstawienie popularnych programów pocztowych....556
Wstępne przygotowanie i konfiguracja....557

Odczytywanie i zapisywanie wiadomości....559
Wysyłanie wiadomości....561
Korzystanie z załączników....562
Przeglądarki internetowe....563
Przedstawienie popularnych przeglądarek internetowych....563
Konfiguracja przeglądarki internetowej....565
Kwestie bezpieczeństwa przeglądarki internetowej....567
Programy klienckie FTP i SFTP....568
Przedstawienie popularnych klientów FTP....569
Podstawowe polecenia FTP....570
FTP w działaniu — transfer plików....573
Klienty współdzielenia plików....575
Użycie eksportowanych katalogów NFS....575
Użycie udziałów SMB/CIFS....576
Klienty zdalnego logowania....580
Użycie klientów Telnetu....580
Użycie klientów SSH....581
Podsumowanie....582

Rozdział 25. Narzędzia biurowe 583

Narzędzia biurowe w systemie FreeBSD....583
GNOME Office....583
KOffice....585
OpenOffice.org....586
Inne narzędzia....587
Problemy z narzędziami biurowymi we FreeBSD....589
Obsługa czcionek....589
Drukowanie....590
Import i eksport plików....591
Przykład pakietu biurowego — OpenOffice.org....592
Konfigurowanie czcionek i drukarek....593
Tworzenie dokumentu....596
Edytowanie tekstu....596
Wykonywanie obliczeń....597
Tworzenie grafiki....599
Tworzenie prezentacji....601
Podsumowanie....603

Rozdział 26. Narzędzia graficzne 605

GIMP....605
Uruchamianie programu GIMP....606
Przedstawienie programu GIMP....607
Ładowanie i zapisywanie plików graficznych....608
Wprowadzanie tekstu i rysowanie....610
Transformacje....613
Narzędzia do edycji grafiki wektorowej....614
Użycie programu Xfig....614
Użycie programu Dia....616
Użycie programu gnuplot....617

Zarządzanie plikami postscriptowymi, EPS i PDF 621
Użycie programu Ghostscript....622
Użycie przeglądarek plików postscriptowych z interfejsem GUI....624
Użycie przeglądarek plików PDF z interfejsem GUI....625
Podsumowanie....626

Rozdział 27. Multimedia i gry 627

Obsługa dźwięku w systemie FreeBSD....627
Obsługiwane karty dźwiękowe 628
Włączenie obsługi dźwięku....629
Narzędzia audio 631
Miksery dźwięku 631
Narzędzia wiersza poleceń do odtwarzania i nagrywania dźwięku 633
Narzędzia GUI do odtwarzania i nagrywania dźwięku....634
Narzędzia do obsługi plików MP3 635
Odtwarzacze płyt CD 638
Aplikacje wideo....639
Najważniejsze formaty plików wideo 639
Odtwarzanie samodzielnych plików wideo....641
Użycie odtwarzacza poprzez przeglądarkę internetową 642
Rozrywka we FreeBSD — gry....644
Tradycyjne gry uniksowe w trybie tekstowym 644
Gry w środowisku X 645
Gry komercyjne....646
Podsumowanie....647

Część VI Obsługa systemu....649

Rozdział 28. Zautomatyzowane i niezautomatyzowane procedury obsługi 651

Czyszczenie plików 651
Pliki dzienników....652
Pliki tymczasowe....654
Nieużywane programy 655
Kontrolowanie plików użytkowników 656
Monitorowanie wykorzystania procesora....658
Narzędzia do kontroli wykorzystania procesora 658
Wykrywanie nadmiernego obciążenia procesora....659
Usuwanie problemów....660
Zapobieganie problemom....661
Monitorowanie wykorzystania pamięci....662
Narzędzia do kontroli wykorzystania pamięci 663
Wykrywanie nadmiernego obciążenia pamięci....664
Zapobieganie i rozwiązywanie problemów....665
Dodawanie przestrzeni wymiany 665
Zachowanie aktualności oprogramowania 668
Znaczenie aktualizacji oprogramowania....668
Narzędzia i procedury do kontrolowania aktualności oprogramowania....668
Podsumowanie....675

Rozdział 29. Kwestie bezpieczeństwa systemu 677
Przegląd metod ataku 677
Rozpoznawanie systemu 678
Ataki zdalne 679
Ataki lokalne 680
Ataki pośrednie 682
Ataki typu Denial-of-Service (DoS) 682
Eliminowanie niepotrzebnego oprogramowania 683
Identyfikowanie nieużywanych programów 684
Ograniczanie dostępu do programów 685
Dezaktywacja programów 686
Usuwanie programów 687
Ograniczenia metod eliminacji oprogramowania 688
Ograniczanie dostępu do serwerów 688
Ochrona serwerów za pomocą haseł 689
Użycie wbudowanych środków zabezpieczających 690
Użycie TCP Wrappers 691
Użycie firewalla 694
Tworzenie bezpiecznych haseł 695
Znaczenie bezpieczeństwa haseł 695
Tworzenie haseł 696
Specjalne uwagi dla konta użytkownika root 698
Wykrywanie próby włamania 698
Ogólne znaki ostrzegawcze 699
Użycie programu Tripwire 700
Inne narzędzia do wykrywania prób włamania 704
Czynności wykonywane po wykryciu włamania 705
Podsumowanie 707

Rozdział 30. Kompilowanie oprogramowania 709
Kiedy należy kompilować oprogramowanie? 709
Zrozumienie procesu kompilacji 710
Zalety i wady samodzielnie skompilowanego oprogramowania 711
Narzędzia do kompilowania oprogramowania 712
Kompilatory języków programowania 712
Biblioteki pomocnicze i pliki nagłówków 713
Narzędzia programistyczne ogólnego przeznaczenia 714
Typowa sesja kompilacji programu 715
Pobranie kodu źródłowego 716
Zapoznanie się z dokumentacją 716
Konfigurowanie programu 717
Kompilowanie programu 718
Instalowanie skompilowanego oprogramowania 719
Sprawdzanie konfliktów pakietów 719
Instalowanie programu 720
Sprawdzanie zainstalowanego pakietu 721
Podsumowanie 722

Rozdział 31. Tworzenie skryptów 723
Programy kompilowane i interpretowane 723
Język maszynowy i kod zrozumiały dla człowieka 724
Kod kompilowany i interpretowany 724
Zalety i wady kodu kompilowanego i interpretowanego 725
Typowe języki skryptowe 726
Rozpoczynanie i kończenie skryptu 727
Użycie poleceń zewnętrznych 728
Uruchamianie programów zewnętrznych 728
Potokowanie programów oraz przekierowywanie wejścia i wyjścia 729
Użycie zmiennych 731
Przypisywanie wartości do zmiennych 731
Użycie argumentów wiersza poleceń jako zmiennych 731
Wyświetlanie i użycie zmiennych 732
Użycie wyrażeń warunkowych 736
Podstawowe narzędzia logiczne 736
Polecenia if, elif i else 737
Polecenie case 740
Warunkowe wykonanie polecenia 741
Użycie pętli 741
Pętla for 742
Pętla while 743
Pętla until 744
Użycie funkcji 745
Podsumowanie 747

Rozdział 32. Rozwiązywanie problemów 749
Źródła pomocy 749
Zasoby pomocy dostępne w systemie 750
Materiały drukowane 751
Zasoby pomocy w Internecie 752
Identyfikowanie problemów sprzętowych i programowych 754
Typowe symptomy problemów sprzętowych 754
Typowe symptomy problemów programowych 755
Procedury diagnozowania problemu 757
Ogólne procedury diagnostyczne 757
Diagnostyka sprzętu 758
Diagnostyka oprogramowania 766
Testowanie rozwiązania 770
Użycie awaryjnego systemu startowego 770
Podsumowanie 772

Dodatki 773

Dodatek A Glosariusz 775

Skorowidz 787

Rozdział 5.
Narzędzia administracyjne

Część II tej książki jest poświęcona kwestiom ***administracji systemem***, czyli zadaniom, które wpływają na działanie systemu operacyjnego i zainstalowane oprogramowanie oraz na konta zwyczajnych użytkowników. Choć opis zadań administracyjnych można znaleźć również w częściach III, IV i VI, kilka kolejnych rozdziałów stanowi wprowadzenie do najważniejszych operacji wykonywanych przez administratora systemu. Typowe czynności wykonywane przez użytkowników, takie jak obsługa edytora tekstu, wykonywanie analizy danych lub pisanie programów, zostaną przedstawione w części V.

W pewnym sensie rozdział 5. stanowi wstęp do wprowadzenia do administracji systemem. Zostaną tu przedstawione podstawowe narzędzia i pliki, które są wykorzystywane przez administratora. Początkowo przyjrzymy się drzewu katalogów */etc*, w którym znajduje się większość plików konfiguracyjnych systemu. Ponieważ większość zadań administratora jest wykonywana poprzez edycję plików tekstowych zapisanych w katalogu */etc*, powinniśmy poznać sposób obsługi co najmniej jednego edytora tekstu. W tym rozdziale omówimy najpopularniejsze programy tego typu (*vi* i gEdit), jednak nie zapomnimy o innych dostępnych narzędziach. Każdy administrator powinien poznać podstawowe polecenia administracyjne, dzięki którym możliwe są: uzyskanie informacji na temat systemu oraz zmiana ogólnych parametrów FreeBSD. Ostatnia część rozdziału zostanie poświęcona narzędziom administracyjnym z graficznym interfejsem użytkownika. Choć wielu tradycjonalistów nie używa takich programów w Uniksie, mogą stać się one przydatne w pewnych sytuacjach, na przykład kiedy mniej doświadczony użytkownik musi wykonać niektóre proste zadania administracyjne.

Katalog /etc i jego zawartość

Większość programów FreeBSD, włącznie z aplikacjami instalowanymi wraz z tym systemem operacyjnym, stosuje się do pewnych konwencji i standardów dotyczących rozmieszczania poszczególnych plików (więcej informacji na temat tych standardów można znaleźć w podrozdziale „Zrozumienie FHS” w rozdziale 7.). Jedna z tych konwencji nakazuje umieszczenie plików konfiguracyjnych systemu w drzewie katalogów */etc*. W większości przypadków nazwa pliku konfiguracyjnego jest zgodna z nazwą samego programu. Bardzo często tworzony jest również specjalny katalog, w którym zapisywane są wszystkie ustawienia. Jednak nie wszystkie pliki konfiguracyjne w */etc* są powiązane z określonym aplikacjami, a ich nazwy mogą wydawać się dość dziwne. Poszczególne pliki z drzewa katalogów */etc* zostaną mniej lub bardziej szczegółowo opisane

w tym rozdziale. Należy jednak pamiętać, że mogą znaleźć się tam również pliki konfiguracyjne programów, które nie zostały przedstawione w tej książce lub które odgrywają poślednią rolę.

Nie wszystkie programy umieszczają pliki konfiguracyjne w */etc*. Pewne aplikacje nie wymagają plików tego typu, podczas gdy inne zapisują je w nietypowych katalogach. Niektóre pliki konfiguracyjne można znaleźć w katalogu */usr/local/etc*, a także w katalogach domowych poszczególnych użytkowników. Są to zwykle pliki, których nazwy rozpoczynają się od kropki (co powoduje ich ukrycie przed poleceniem `ls`), zaś po kropce występuje nazwa docelowego programu.

Pliki konfiguracyjne */etc* można podzielić na trzy grupy: pliki startowe systemu, które decydują o procesie uruchomienia FreeBSD po załadowaniu jądra; pliki konfiguracyjne serwerów, które kontrolują działanie poszczególnych programów serwerowych; oraz pozostałe pliki konfiguracyjne, zawierające ustawienia programów użytkownika oraz ogólne ustawienia systemowe.

Pliki startowe systemu

W rozdziale 3. przedstawiono opis wieloetapowego procesu startowego FreeBSD. Po uruchomieniu komputera wykonywany jest kod zapisany w BIOS-ie, po czym następuje uruchomienie programu rozruchowego i załadowanie jądra FreeBSD. Jądro uruchamia następnie program */sbin/init*, który kontroluje dalsze etapy ładowania systemu, czyli procesy działające już wewnątrz FreeBSD.

We FreeBSD (podobnie jak w wielu systemach operacyjnych) program musi być uruchamiany przez inny program, który jest nazywany ***rodzicem*** dla pierwszego programu, czyli ***potomka***. Działający program nosi nazwę ***procesu***, a każdy proces otrzymuje własny ***identyfikator procesu*** (***PID***), czyli numer powiązany z danym procesem. Tworzenie nowego procesu jest często nazywane ***rozmnażaniem procesów***.

Relacje pomiędzy rodzicami, potomkami oraz identyfikatorami PID odnoszą się również do plików startowych systemu, ponieważ proces *init* jest rodzicem (lub bardziej odległym przodkiem) wszystkich innych procesów. Proces *init* jest wywoływany przez jądro natychmiast po uruchomieniu się. Plik konfiguracyjny *init* nosi nazwę */etc/rc* (tak naprawdę jest to skrypt powłoki, gdyż proces *init* uruchamia powłokę */bin/sh*, w której wywoływany jest skrypt */etc/rc*). Oznacza to, że proces startowy systemu oraz wszystkie jego procesy są kontrolowane właśnie przez */etc/rc*. Modyfikacja skryptu */etc/rc* lub jego podrzędnych plików może wpłynąć na zachowanie FreeBSD.

Proces rozruchowy FreeBSD nosi ogólną nazwę ***metody startowej BSD***. Inne systemy uniksowe, takie jak większość dystrybucji Linuksa, wykorzystują inną metodę, zwaną ***skryptami startowymi SysV***. Metoda BSD jest związana z użyciem kilku dużych skryptów, podczas gdy metoda SysV wymaga znacznej liczby małych skryptów. Metoda SysV zapewnia także funkcję znaną jako ***poziomy działania***; każdy poziom działania obsługuje inny zestaw uruchomionych programów serwerowych i innych procesów. Taka funkcja nie jest jednak dostępna w metodzie startowej BSD, a zarazem we FreeBSD.

Skrypt */etc/rc* wykonuje wiele operacji, takich jak sprawdzenie poprawności dysków, usunięcie starych plików z drzewa katalogów */var* oraz aktywacja partycji wymiany. Duża liczba zadań jest przekazywana do innych procesów poprzez wywoływanie skryptów zapisanych w */etc*, których nazwy rozpoczynają się od *rc* (na przykład *rc.serial* i *rc.network*). Wszystkie te pliki kontrolują proces uruchamiania FreeBSD i stanowią najważniejsze pliki konfiguracyjne systemu.

Skrypty startowe FreeBSD są zwyczajnymi skryptami. Umiejętność tworzenia i edycji skryptów powłoki (zobacz rozdział 31.) pozwoli na dokonanie zarówno prostych, jak i bardziej złożonych zmian w konfiguracji FreeBSD. Takie zmiany można również wykonać poprzez odnalezienie wywołania do programu, który kontroluje żądaną opcję, a następnie modyfikację tego wywołania. W tym kontekście słowo ***wywołanie*** oznacza odniesienie powodujące uruchomienie danego programu. Bardzo często jest to po prostu nazwa programu umieszczona w jednym ze skryptów.

Modyfikacja skryptów startowych FreeBSD jest bardzo niebezpieczną operacją. Nawet drobna pomyłka może całkowicie uniemożliwić uruchomienie systemu. Co gorsza, błąd może ujawnić się dopiero po kilku tygodniach lub miesiącach od momentu dokonania zmiany, jeżeli komputer nie zostanie natychmiast zresetowany. Przed dokonaniem jakiejkowiek zmiany skryptów startowych należy utworzyć ich kopię zapasową, a po zakończeniu edycji trzeba natychmiast zrestartować komputer. Jeżeli system nie uruchomi się, należy użyć dyskietki ratunkowej, aby przywrócić oryginalny plik.

Zmiana procedury startowej FreeBSD odbywa się także poprzez modyfikację dwóch innych plików lub klas plików, a mianowicie skryptu */etc/rc.local* i skryptów w katalogu */usr/local/etc/rc.d*. Choć obie klasy plików mogą być zmienione przez użytkownika, w przypadku nowszych wersji FreeBSD zaleca się użycie skryptów w tym drugim katalogu. W rozdziale 6. znajduje się szczegółowy opis sposobu modyfikacji tych skryptów.

Nazwy pliku */etc/rc.local* i katalogu */usr/local/etc/rc.d* zawierają słowo *local*, co wskazuje, że poszczególne skrypty i programy odnoszą się do konkretnego systemu. Nie są to standardowe narzędzia, które można znaleźć we wszystkich systemach FreeBSD.

Pliki konfiguracyjne programów serwerowych

Drugą grupę plików konfiguracyjnych w drzewie katalogów */etc* stanowią pliki związane z programami serwerowymi. Większość tych plików będzie omówiona bardziej szczegółowo w kolejnych rozdziałach tej książki, a w szczególności w części III. Poniżej przedstawiono jedynie krótkie opisy najważniejszych plików konfiguracyjnych i ich dokładnego położenia w drzewie katalogów:

- **Pliki konfiguracyjne X**. Plik */etc/XF86Config* i katalog */etc/X11* sterują działaniem serwera XFree86 i powiązanych programów, takich jak graficzne narzędzie do logowania *X Display Manager* (*XDM*). Plik *XF86Config* zostanie opisany w rozdziale 13.
- **Pliki konfiguracyjne Apache**. Katalog */usr/local/etc/apache* zawiera pliki z konfiguracją popularnego serwera WWW o nazwie Apache. Największą

rolę odgrywa plik *httpd.conf*. Pozostałe pliki w tym katalogu są plikami pomocniczymi. Opis konfiguracji Apache znajduje się w rozdziale 20.

- **Plik *./etc/exports***. Ten plik zawiera listę katalogów, które FreeBSD udostępnia poprzez serwer plików *NFS* (ang. *Network File System*); serwer ten zostanie przedstawiony w rozdziale 18.
- **Pliki TCP Wrappers**. Program TCP Wrappers to bardzo ważny pakiet do obsługi zabezpieczeń, który jest wykorzystywany przez wiele programów serwerowych. Ustawienia programu są zapisane w plikach */etc/hosts.allow* i */etc/hosts.deny*. Opis pakietu można znaleźć w rozdziale 29.
- **Pliki dostępu hostów**. Plik */etc/hosts.equiv* jest używany przez wiele programów serwerowych i zawiera listę zdalnych komputerów, które mogą uzyskać dostęp do serwerów lokalnych. Podobną rolę odgrywa plik */etc/hosts.lpd*, chociaż jest wykorzystywany tylko przez standardowy serwer druku we FreeBSD.
- **Plik *./etc/inetd.conf***. Ten plik odgrywa ogromną rolę, ponieważ steruje działaniem ***superserwera*** FreeBSD o nazwie *inetd*. Ten program zastępuje wiele innych serwerów, uruchamiając niezbędne programy dopiero w razie potrzeby. Pozwala to znaczne zmniejszenie wykorzystania pamięci w komputerze, który obsługuje rzadziej używane programy serwerowe. *inetd* używa TCP Wrappers w celu zapewnienia wspólnych mechanizmów kontroli dostępu dla wszystkich obsługiwanych serwerów. Więcej informacji na temat konfiguracji *inetd.conf* można znaleźć w podrozdziale „Użycie superserwera” w rozdziale 19.
- **Katalog *./etc/mail***. W tym katalogu można znaleźć wiele plików z ustawieniami programu *sendmail*, czyli standardowego serwera *SMTP* (ang. *Simple Mail Transfer Protocol*). Plik */etc/mail/sendmail.cf* to główny plik konfiguracyjny *sendmail*, który jest generowany na podstawie pliku */etc/mail/freebsd.mc*, co opisano w rozdziale 19.
- **Plik *./usr/local/etc/smb.conf***. Jest to plik służący do konfiguracji Samby, czyli serwera plików i druku dla klientów Microsoft Windows. W czasie instalacji Samby tworzony jest plik */usr/local/etc/smb.conf.default*, który należy skopiować do */usr/local/etc/smb.conf* i zmodyfikować w sposób opisany w rozdziale 18.
- **Pliki SSH**. Pakiet *SSH* (ang. *Secure Shell*) zapewnia funkcje zdalnego logowania poprzez połączenie szyfrowane. Katalog */etc/ssh* zawiera pliki służące do konfiguracji zarówno klientów (*ssh_config*), jak i serwera (*sshd_config*).

Powyższa lista nie jest wyczerpująca, gdyż na komputerach z systemem FreeBSD używanych jest wiele programów serwerowych, a w tej książce brak miejsca na przedstawienie nawet ich części. Pliki konfiguracyjne dodatkowych serwerów są umieszczane zwykle w katalogach */etc* lub */usr/local/etc*. Dodatkowe informacje można znaleźć w dokumentacji programu. Na liście znalazły się również pliki konfiguracyjne, które nie są obecne na każdym komputerze, ponieważ poszczególni użytkownicy mogą wybrać inny zestaw oprogramowania. Więcej szczegółów na temat instalacji serwerów przedstawiono w rozdziale 11.

Aby uruchomić program serwerowy, zwykle należy dodać odpowiedni plik do katalogu */usr/local/etc/rc.d* (zobacz podrozdział „Pliki startowe systemu”) lub umieścić nowy wpis w pliku */etc/inetd.conf* (zostanie on opisany w rozdziale 16.).

Inne pliki konfiguracyjne

We wcześniejszej części rozdziału przedstawiono pliki konfiguracyjne, które są powiązane ze skryptami startowymi lub określonymi serwerami. Inne pliki tego typu służą do ustawiania ogólnych parametrów systemowych lub są luźno związane z niektórymi programami serwerowymi lub procedurami startowymi. Przykładem takich plików konfiguracyjnych są ustawienia programu TCP Wrappers. Poniżej przedstawiono inne pliki tego typu:

- **Plik *`/etc/crontab`***. Wraz z FreeBSD instalowane jest narzędzie o nazwie *crontab*, które służy do uruchamiania programów o określonej godzinie. Pozwala to na automatyczne wykonywanie określonych procedur obsługi i naprawy systemu (zobacz rozdział 28.). Plik */etc/crontab* zawiera listę wszystkich zadań programu *crontab*. Format tego pliku przedstawiono w rozdziale 6.
- **Plik *`/etc/dhclient.conf`***. Niektóre komputery używają programu klienckiego *dhclient* w celu automatycznego uzyskania adresu IP i powiązanych informacji o sieci z serwera *DHCP* (ang. *Dynamic Host Configuration Protocol*). Ustawienia *dhclient* są zapisywane w pliku */etc/dhclient.conf*. W większości przypadków nie ma konieczności modyfikacji tych ustawień. Więcej szczegółów na ten temat można znaleźć w rozdziale 14.
- **Plik *`/etc/fstab`***. Ten plik odgrywa ogromną rolę przy zarządzaniu dyskami twardymi i innymi nośnikami, gdyż umieszczane są w nim informacje dotyczące poszczególnych partycji, systemów plików oraz punktów montowania w drzewie katalogów FreeBSD. Aby uzyskać dodatkowe informacje, należy przejść do rozdziału 7.
- **Plik *`/etc/printcap`***. Funkcje drukowania we FreeBSD są zależne od pliku o nazwie */etc/printcap*, który zawiera listę drukarek, opis portów lub serwerów służących do łączenia się z drukarkami oraz informacje o programach wykorzystywanych do przetwarzania zadań listingu. W rozdziale 9. można znaleźć więcej szczegółów na ten temat.
- **Plik *`/etc/resolv.conf`***. Ten plik jest wykorzystywany do zdefiniowania serwerów nazw, które będą wykorzystywane przez FreeBSD do konwersji nazw hostów do postaci adresów IP. Szczegółowe informacje znajdują się w rozdziale 14.
- **Pliki użytkowników i grup**. Pliki */etc/passwd*, */etc/master.passwd* i */etc/group* zawierają dane dotyczące kont użytkowników i grup. Są one modyfikowane w czasie zmiany ustawień kont użytkowników, co opisano w rozdziale 10. Do tego celu zaleca się jednak użycie specjalnych narzędzi, gdyż bezpośrednia edycja plików konfiguracyjnych w edytorze tekstu może spowodować powstanie problemów.

Również w tym przypadku lista plików konfiguracyjnych nie jest kompletna. Nie wszystkie pliki są obecne w każdym komputerze, ale większość z nich jest tworzona domyślnie. Inne pliki mogą być używane tylko w określonych konfiguracjach systemu.

Przegląd edytorów tekstu we FreeBSD

Zanim rozpoczniemy edycję plików konfiguracyjnych w celu dopasowania ustawień systemowych, musimy poznać sposób wykonania tej operacji. W tej części rozdziału przedstawione są niektóre edytory tekstu instalowane wraz z FreeBSD oraz przykładowe sesje w dwóch programach tego typu.

Edytory tekstu można podzielić na wiele różnych kategorii, ale dwie najważniejsze grupy to edytory tekstu, działające w trybie tekstowym, i edytory z graficznym interfejsem użytkownika. Nawet osoby pracujące w środowisku graficznym powinny poznać co najmniej jeden edytor pracujący w trybie tekstowym, co pozwoli na rozwiązanie wielu problemów w sytuacji awaryjnej. Niektórzy użytkownicy wolą korzystać z takich narzędzi również w środowisku graficznym.

Należy odróżnić edytory tekstu od procesorów tekstowych. Edytory tekstu służą do modyfikacji plików tekstowych, które nie są zwykle formatowane w żaden sposób i nie zawierają różnych czcionek. Takie narzędzia są wykorzystywane najczęściej do edycji plików konfiguracyjnych i otwierania dokumentacji oprogramowania. Procesory tekstowe są używane do tworzenia dokumentów, które będą drukowane lub wymagają określonego sposobu formatowania. Możliwe jest jednak użycie edytora tekstu w połączeniu ze specjalnym programem do formatowania, takim jak TeX. Niektórzy użytkownicy preferują użycie takiego zestawu narzędzi, a nie typowego procesora tekstu. Więcej informacji na temat przetwarzania tekstu we FreeBSD można znaleźć w rozdziale 25.

Edytory tekstu pracujące w trybie tekstowym

Edytory tekstu należące do tej grupy zapewniają, w pewnym sensie, największą funkcjonalność, gdyż możliwe jest ich uruchomienie zarówno w środowisku czysto tekstowym, jak i w oknie *xterm* w środowisku graficznym. Teoretycznie oznacza to, iż poznanie jednego edytora pozwoli na wykonanie wszystkich zadań związanych z edycją tekstu we FreeBSD. Nie jest to do końca jednak prawdą, gdyż do wykonania niektórych operacji domyślnie wybierane są inne narzędzia, a w sytuacji awaryjnej nie zawsze będziemy mieli dostęp do ulubionego edytora. Dlatego też należy poznać (chociaż pobieżnie) najbardziej popularne edytory tekstu. Przedstawiamy je poniżej:

- **vi**. Historia tego edytora sięga początków Uniksa. Program posiada nietypowy interfejs użytkownika z trzema typami pracy, który dla wielu osób jest zbyt zagmatwany. Pomimo tych niedogodności *vi* zyskał ogromną popularność, głównie dzięki dużej elastyczności i relatywnej prostocie. *vi* jest niewielkim programem, przez co często jest umieszczany na dyskietkach ratunkowych, które są wykorzystywane w przypadku awarii systemu. Szczegółowy opis pracy w *vi* przedstawiono w podrozdziale „Przykład — edycja pliku w vi". Warto wiedzieć, że *vi* to tak naprawdę mała rodzina edytorów, gdyż dostępne są różne warianty, takie jak Vim i Elvis. Z punktu widzenia początkującego użytkownika wszystkie te edytory są jednak identyczne.
- **Emacs**. Ten edytor stanowi całkowite przeciwieństwo *vi*, gdyż Emacs jest bardzo dużym i skomplikowanym programem. W środowisku użytkowników

Uniksa krąży dowcip, że Unix służy do uruchamiania programów, a Emacs jest systemem operacyjnym — i wszystkim innym. Emacs może być wykorzystany nie tylko do edycji plików tekstowych, ale również jako zintegrowane środowisko programowania (IDE), czytnik grup dyskusyjnych, klient pocztowy, a nawet jako przeglądarka internetowa. Program jest dostępny zarówno w „oryginalnej" postaci, jak i w wersji graficznej o nazwie XEmacs. Pełny opis tego edytora wymagałby książki o podobnej objętości jak ta.

- **Niewielkie klony Emacs**. Użytkownicy programu Emacs często potrzebują mniejszego narzędzia, które można umieścić na przykład na dyskietce ratunkowej. Dostępnych jest wiele klonów tego edytora o okrojonych funkcjach, na przykład *jed*, *joe*, *jove* i *uemacs* (nazywany również micro-Emacs). Większość tych edytorów jest instalowanych wraz z FreeBSD, co pozwala na ich użycie zamiast (lub oprócz) edytora *vi*.
- **Easy Editor**. Ten niewielki edytor tekstów jest używany do edycji plików w czasie instalacji FreeBSD. W rozdziale 2. umieszczono krótki opis pracy w tym programie. Aby uruchomić Easy Editor, należy wpisać `ee` oraz (opcjonalnie) nazwę pliku tekstowego, który będzie edytowany.

Oczywiście powyższa lista nie jest wyczerpująca, gdyż programiści przygotowali wiele różnych edytorów tekstu dla systemów uniksowych. Większość z nich jest dostępnych w menu *Editors* narzędzia *sysinstall*, które pozwala na instalację dodatkowych pakietów oprogramowania. Należy jednak wspomnieć, że w menu *Editors* dostępne są edytory pracujące w trybie tekstowym i graficznym, jak również dodatkowe moduły dla tych narzędzi. Nie zawsze jest więc jasne, które pakiety należy umieścić na dysku twardym.

Warto w tym miejscu wspomnieć o pewnej kategorii programów, które nie są edytorami tekstu sensu stricte. Są to tak zwane ***pagery*** (***programy stronicowania***), które strona po stronie wyświetlają na ekranie pliki tekstowe. Pierwszy pager uniksowy nosi nazwę *more* i jest dostępny również we FreeBSD. Pager *more* jest wykorzystywany przez polecenie `man` (zobacz opis w dalszej części rozdziału) do wyświetlania dokumentacji systemowej. To narzędzie jest jednak dość ograniczone i coraz częściej jest zastępowane przez pager o nazwie *less*.

Aby użyć pager, należy wpisać jego nazwę oraz nazwę pliku tekstowego, na przykład `less plik.txt`. Na ekranie pojawi się pierwsza strona tego pliku. Wciśnięcie spacji powoduje przejście do kolejnej strony, natomiast klawisze strzałek przewijają ekran o jeden wiersz. Wciśnięcie klawiszy *Esc* i < powoduje przejście do początku pliku, natomiast kombinacja klawiszy *Esc* i > powoduje skok do końca pliku (w przypadku pagera *more* spowoduje to także wyjście z programu). Aby przeszukać plik, należy wpisać ukośnik (/), a bezpośrednio po nim szukaną frazę. Poszczególne pagery (a w szczególności pager *less*) obsługują dodatkowe funkcje. Aby je poznać, należy użyć polecenia `man`, na przykład `man more` lub `man less`.

Edytory tekstowe z graficznym interfejsem użytkownika

Osoby migrujące do FreeBSD z systemów Windows lub Mac OS często wybierają narzędzia z graficznym interfejsem użytkownika. Na szczęście, we FreeBSD dostępnych

jest wiele edytorów tekstu pracujących w środowisku graficznym, dzięki czemu każdy może dopasować odpowiedni program do swoich potrzeb. Poniżej przedstawiono kilka przykładowych edytorów tekstu z interfejsem GUI:

- **Emacs**. Jedną z wielu funkcji edytora Emacs jest obsługa środowiska X, choć pierwsze wersje tego narzędzia miały z tym pewne problemy. Program może być uruchomiony we własnym oknie i zapewnia kilka menu, ale w dalszym ciągu nie jest w pełni zintegrowany z X. Osoby preferujące pracę w tym edytorze powinny zapoznać się z jego graficzną odmianą o nazwie XEmacs. Warto wiedzieć, że XEmacs pracuje także w czystym trybie tekstowym.
- **NEdit**. Ten edytor (*http://nedit.org*) powstał z myślą o dotychczasowych użytkownikach systemów Windows i Mac OS. Choć nie jest tak rozbudowany jak Emacs, zapewnia dużą liczbę funkcji, które mogą być wykorzystane zarówno w czasie programowania, jak i w czasie skomplikowanej edycji dużej liczby plików tekstowych.
- **gEdit**. Jest to standardowy edytor tekstu w *GNOME* (ang. *GNU Network Object Model Environment*), dzięki czemu można go uruchomić bezpośrednio z menu tego środowiska. gEdit zapewnia jedynie podstawowe funkcje edycji tekstu. Ponieważ jest on instalowany domyślnie wraz z wieloma wersjami GNOME, jego szczegółowy opis zostanie przedstawiony w podrozdziale „Przykład — edycja pliku w gEdit".
- **xedit**. Jest to dość prosty edytor tekstu pracujący w środowisku graficznym. Choć nie zapewnia tak dużej liczby funkcji, jak Emacs lub NEdit, dzięki małej wielkości jest często używany na komputerach z okrojoną wersją X.

Również powyższa lista edytorów tekstu nie zawiera wszystkich dostępnych narzędzi. Warto wiedzieć, że niektóre edytory działające w trybie tekstowym (takie jak *vi* i niektóre klony Emacs) posiadają rozszerzenia umożliwiające pracę w X. Ich funkcje są jednak dość ograniczone, gdyż w większości przypadków pozwalają na otwarcie tylko kilku okien i sterowanie poprzez pasek menu. Takie rozszerzenia nie korzystają z okien dialogowych, czcionek i innych elementów X. Duża liczba edytorów tekstu z graficznym interfejsem użytkownika jest dostępna w menu *Editors* programu *sysinstall*. Inne narzędzia tego typu można znaleźć w Internecie.

Przykład — edycja pliku w vi

Jak już wspomniano, edytor tekstu *vi* zapewnia ogromną liczbę funkcji, ale jego obsługa może być skomplikowana dla początkujących użytkowników, którzy są przyzwyczajeni do bardziej intuicyjnych narzędzi. Jest to spowodowane głównie obecnością trzech różnych ***trybów pracy***. Znaczenie poszczególnych klawiszy zmienia się w zależności od wybranego trybu *vi*. Edytor *vi* pozwala na wykonanie typowych operacji, takich jak wprowadzanie, usuwanie i wklejanie tekstu. Oczywiście możliwe jest także otwieranie i zapisywanie plików tekstowych. W tej części rozdziału zapoznamy się jednak tylko z podstawowymi funkcjami *vi*, gdyż ich pełny opis wymagałby napisania zupełnie innej książki.

Użycie trybów vi

W każdym z trzech trybów pracy *vi* dostępne są inne funkcje edycyjne. Poniżej przedstawiono krótki opis każdego trybu:

- **Tryb poleceń (command mode)**. W tym trybie należy wprowadzać polecenia służące do manipulacji tekstem lub wykonywania innych operacji. Większość poleceń można wywołać, wciskając pojedyncze klawisze, na przykład polecenie `o` tworzy nowy wiersz poniżej bieżącego wiersza. Do nawigacji po pliku w trybie poleceń można użyć klawiszy strzałek na klawiaturze.
- **Tryb rozszerzony (ex mode)**. Ten tryb umożliwia wykonanie operacji na zewnętrznych plikach. Pozwala to na przykład na uruchomienie zewnętrznych programów wykonujących określoną czynność, taką jak sprawdzenie pisowni lub sformatowanie kodu źródłowego w standardowy sposób. Tryb rozszerzony jest używany również do zapisywania plików, co zostanie opisane w podrozdziale „Zapisywanie zmian". Aby użyć tego trybu, należy przejść do trybu poleceń i wpisać dwukropek (`:`) oraz wybrane polecenie trybu rozszerzonego.
- **Tryb edycji (edit mode)**. Praca w tym trybie przypomina obsługę innych edytorów tekstu, gdyż na ekranie pojawiają się wszystkie znaki wprowadzane na klawiaturze. Wciśnięcie klawisza *Esc* powoduje powrót do trybu poleceń.

Jeżeli użytkownik nie wie, w jakim trybie znajduje się program *vi*, powinien wcisnąć klawisz *Esc*. Spowoduje to natychmiastowe przejście do trybu poleceń.

Powyższy opis trzech trybów pracy *vi* może wydawać się dość atrakcyjny. Zapoznanie się z przedstawionym tu przykładem pozwoli jednak zrozumieć różnice pomiędzy poszczególnymi trybami *vi*.

Edycja pliku

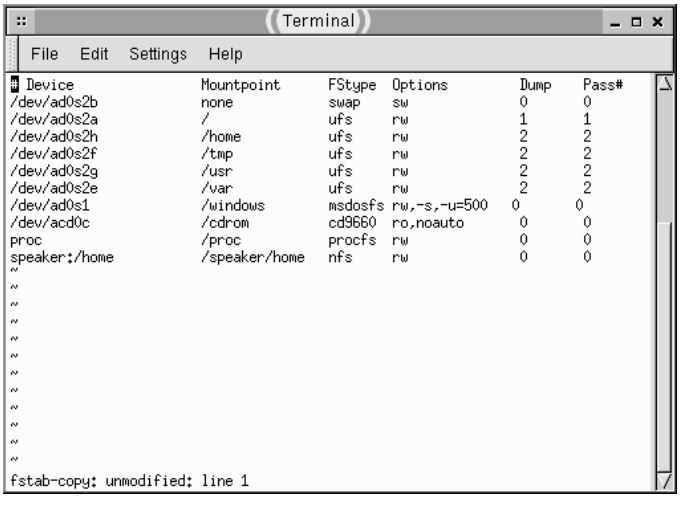
Przed rozpoczęciem pracy w *vi* należy przygotować plik do edycji. W praktyce może to być dowolny plik. W tym przykładzie użyjemy pliku */etc/fstab*, który we FreeBSD służy do odwzorowania partycji do katalogów w drzewie katalogów FreeBSD. Przedstawione tu operacje pozwolą na dodanie i usunięcie niektórych wpisów w tym pliku. Pozwoli to na dodanie do systemu nowych partycji lub nośników wymiennych, jak również na zablokowanie dostępu do aktywnych partycji. Ponieważ jest to tylko ćwiczenie, należy utworzyć kopię edytowanego pliku. Aby to zrobić, trzeba przejść do katalogu domowego i wpisać polecenie:

```
$ cp /etc/fstab ./fstab-copy
```

Wykonując to ćwiczenie nie należy edytować pliku */etc/fstab*, gdyż może to spowodować problemy z uruchomieniem FreeBSD. Zalecane jest skopiowanie tego pliku do katalogu domowego i edycja kopii. Tę operację należy wykonać jako zwyczajny użytkownik (a nie administrator), dzięki czemu istnieje mniejsze prawdopodobieństwo przypadkowego zapisania zmian w oryginalnym pliku.

Po skopiowaniu pliku możemy rozpocząć jego edycję. W tym celu powinniśmy wykonać następujące czynności:

1. Wpisując polecenie `vi fstab-copy`, musimy uruchomić *vi* i załadować plik *fstab-copy* do buforu *vi*. Rysunek 5.1 przedstawia wynik tej operacji wykonanej w edytorze *vi* uruchomionym w oknie terminala GNOME. Jeżeli użyto odmiennej wersji *xterm*, na ekranie będą widoczne inne ramki i czcionki. W czasie pracy w trybie tekstowym system wyświetli oczywiście tylko tekst. Znaki tyldy (~), widoczne po lewej stronie okna, są używane przez *vi* do oznaczenia końca pliku. W dolnym wierszu znajduje się pasek statusu. Oczywiście plik *fstab-copy* w systemie Czytelnika będzie różnił się od takiego samego pliku z rysunku 5.1. W tym momencie nie ma to praktycznego znaczenia, gdyż naszym celem jest poznanie sposobu edycji plików tekstowych.



Rysunek 5.1. *vi przypomina każdy inny edytor tekstu. Jego unikalne funkcje są dostępne po wprowadzeniu specjalnych poleceń*

2. Załóżmy, że należy usunąć jeden z wpisów w tym pliku, na przykład wpis dla katalogu */windows*. Aby to zrobić, trzeba przejść do tego wiersza poprzez wciśnięcie siedem razy klawisza strzałki w dół. Wskaźnik powinien znaleźć się na początku wiersza */dev/ad0s1*.

3. Polecenie służące do usuwania tekstu jest wprowadzane w trybie poleceń, czyli w domyślnym trybie pracy *vi*. Należy wpisać `dd`, aby usunąć wiersz wskazywany przez wskaźnik. Aby szybko usunąć kilka wierszy, trzeba poprzedzić polecenie `dd` liczbą wierszy do usunięcia, na przykład `3dd` spowoduje zniknięcie trzech wierszy tekstu.

4. Teraz utwórzmy nowy wpis w bieżącym położeniu wskaźnika. Najpierw należy przygotować pusty wiersz. Do tego celu służy polecenie `o`, które należy użyć w trybie poleceń *vi*. Spowoduje to utworzenie pustego wiersza i przeniesienie wskaźnika.

5. Aby wpisać tekst, należy użyć trybu edycji. Polecenie `o`, które zostało użyte w poprzednim punkcie, powoduje automatycznie przejście do tego trybu. Aby przejść do tego trybu z trybu poleceń, należy wpisać `i`. Od tej chwili możliwe jest wprowadzanie tekstu z klawiatury. Użycie polecenia `R` zamiast `i` umożliwia zastępowanie już istniejącego tekstu. Polecenia `a` powoduje przejście do trybu edycji i przesunięcie wskaźnika o jedną pozycję.

6. Teraz możemy wprowadzić nowy wiersz, na przykład:

```
/dev/da0s4      /zip    msdosfs     rw,noauto     0     0
```

7. Wciśnięcie klawisza *Esc* pozwala na powrót do trybu poleceń.
8. Wyobraźmy sobie, że konieczne jest dodanie nowego wiersza, który będzie odpowiednikiem poprzedniego wpisu, ale pozwoli FreeBSD na montowanie dysków Zip z systemem plików FFS, a nie FAT. Pierwszym działaniem będzie wpisanie `yy`; jest to polecenie `yank`, które służy do kopiowania tekstu do wewnętrznego buforu. Polecenie `yy` kopiuje jeden wiersz tekstu. Aby skopiować kilka wierszy, należy podać ich liczbę, na przykład `4yy` umieszcza w buforze cztery wiersze.
9. Teraz należy wpisać polecenie `p`, co spowoduje wklejenie tekstu z wewnętrznego buforu. Na ekranie powinny być widoczne dwa nowe, identyczne wiersze.
10. Używając klawiszy strzałek, trzeba umieścić wskaźnik na spacji tuż po słowie */zip* w jednym z dwóch wierszy.
11. Wpisanie `R` pozwoli na edycję tekstu w trybie zastępowania.
12. Należy wpisać *`-ffs`*, aby uzyskać ciąg */zip-ffs* zamiast */zip*.
13. Teraz należy wcisnąć klawisz *Esc*, aby powrócić do trybu poleceń, a następnie umieścić wskaźnik na literze *m* w słowie *msdosfs* w modyfikowanym wierszu.
14. Wpisanie `R` pozwoli na edycję tekstu w trybie zastępowania.
15. Należy wpisać *`ufs`* i cztery spacje, aby zastąpić ciąg *msdosfs* przez *ufs*.

Wykonanie powyższej procedury spowodowało zastąpienie wiersza */windows* dwoma wierszami do obsługi dysków Zip. Pozostałe wiersze w pliku pozostały niezmienione. Zalecam wykonanie dodatkowych operacji dodawania, usuwania i modyfikacji tekstu w *vi*, co pozwoli nabrać biegłości niezbędnej w czasie edycji plików tekstowych. Poniżej przedstawiono inne funkcje do manipulacji tekstem w *vi*:

- **Cofnięcie operacji**. W edytorze *vi* dostępna jest funkcja *undo*, która pozwala na odwołanie ostatniej operacji. Aby to zrobić, należy użyć polecenia `u`. Ponieważ obsługiwany jest tylko jeden poziom *undo*, ponowne wpisanie `u` spowoduje przywrócenie odwołanej zmiany.
- **Wyszukiwanie tekstu**. Wciśnięcie klawisza ukośnika (*/*) w trybie poleceń rozpoczyna wyszukiwanie tekstu. Wpisanie polecenia `/cd9660` i wciśnięcie klawisza *Enter* w czasie edycji pliku z rysunku 5.1 spowoduje umieszczenie wskaźnika na literze *c* w wyrazie *cd9660*. Znak zapytania (`?`) rozpoczyna wyszukiwanie w odwrotnym kierunku.
- **Zastępowanie tekstu**. Funkcja globalnego zastępowania tekstu powoduje zastąpienie wszystkich wystąpień określonej frazy w pliku tekstowym inną frazą. Aby to zrobić, należy przejść do trybu poleceń i wpisać polecenie `:%s/`*`stary_tekst`*`/`*`nowy_tekst`*, gdzie *`stary_tekst`* to fraza do zastąpienia, a *`nowy_tekst`* to oczywiście nowa fraza. Wciśnięcie klawisza *Enter* spowoduje wykonanie operacji zastępowania.

Więcej szczegółów na temat poleceń `vi` można znaleźć w dokumentacji programu. Dobrym źródłem informacji jest również strona Vi IMproved znajdująca się pod adresem *http://www.vim.org*. Osoby często korzystające z tego edytora tekstu powinny rozważyć zakup książki poświęconej *vi*.

Zapisywanie zmian

Przed zamknięciem programu *vi* należy pamiętać o zachowaniu wszystkich dokonanych zmian. Do tego celu można wykorzystać wiele poleceń trybu rozszerzonego. Przedstawiamy je poniżej:

- **Polecenie** `:w` powoduje zapisanie wszystkich zmian. Jeśli nie podano żadnych opcji, zmiany zostaną zachowane w oryginalnym pliku. Aby utworzyć nowy plik, należy dodać jego nazwę do polecenia `:w`, na przykład polecenie `:w nowy-plik.txt` spowoduje zapisanie dokumentu tekstowego w pliku *nowy-plik.txt*.
- **Polecenie** `:e` może być użyte, jeżeli *vi* został uruchomiony bez podania nazwy pliku lub jeśli konieczne jest utworzenie nowego pliku, na przykład polecenie `:e /etc/printcap` spowoduje otwarcie pliku */etc/printcap*. Należy jednak pamiętać, że *vi* pozwala na jednoczesną edycję tylko jednego pliku, przez co może być konieczne zamknięcie poprzedniego dokumentu.
- **Polecenie** `:r` działa podobnie jak `:e`, ale nowy plik nie zastępuje już istniejącego. To polecenie łączy dwa pliki i zapisuje je przy użyciu nazwy pierwszego pliku, co w niektórych przypadkach jest bardzo wygodne.
- **Polecenie** `:q` powoduje wyjście z programu *vi*.

W jednym poleceniu można połączyć nawet kilka powyższych poleceń, na przykład polecenie `:wq` jest często używane do zapisania wszystkich zmian i zamknięcia *vi*.

Od czasu do czasu może być konieczne użycie specjalnego przełącznika w celu pominięcia wszystkich dokonanych zmian. Taki przełącznik ma postać wykrzyknika (`!`) i występuje zwykle w połączeniu z poleceniami `:e` i `:q`. Wpisanie polecenia `:q!` spowoduje wyjście z *vi* bez zapisywania zmodyfikowanego pliku. Będzie to przydatne, jeżeli użytkownik popełnił błąd lub edytuje niewłaściwy plik tekstowy.

Przykład — edycja pliku w gEdit

Edytory tekstu pracujące w trybie tekstowym w zupełności wystarczają do edycji wszystkich plików konfiguracyjnych FreeBSD, ale niektóre osoby wybierają programy z graficznym interfejsem użytkownika. Narzędzia tego typu zapewniają większą kontrolę nad użytymi czcionkami, kolorami i innymi elementami interfejsu, a pasek menu i zawarte w nim polecenia znacznie ułatwiają wykonanie niektórych operacji. Poszczególne edytory tekstu mogą znacznie się różnić, ale gEdit jest typowym przedstawicielem tej kategorii oprogramowania. gEdit jest również instalowany wraz z większością systemów FreeBSD, przez co warto zapoznać się ze sposobem jego obsługi.

Edycja pliku

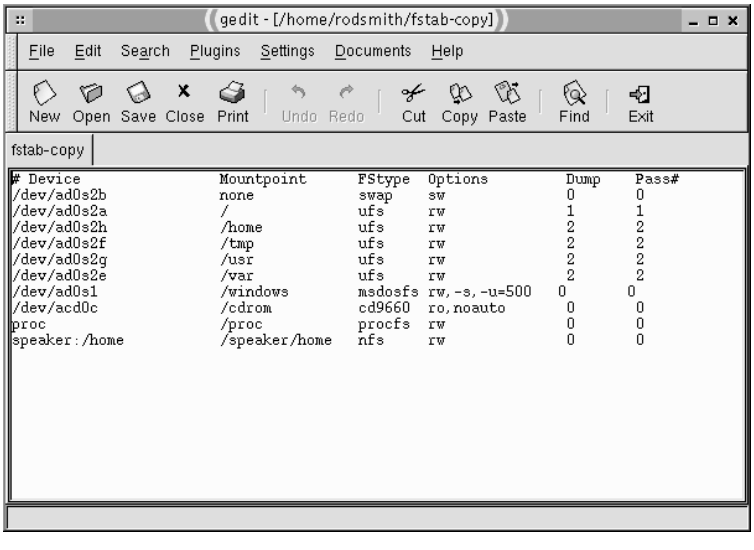
Podobnie jak w poprzednim ćwiczeniu, również w tym przykładzie użyjemy kopii pliku */etc/fstab*. Poniższe polecenie spowoduje skopiowanie tego pliku do katalogu domowego:

```
$ cp /etc/fstab ./fstab-copy
```

Podczas wykonywania tego ćwiczenia nie należy edytować pliku */etc/fstab*, gdyż może to spowodować problemy z uruchomieniem FreeBSD. Zalecane jest skopiowanie tego pliku do katalogu domowego i edycja kopii. Tę operację należy wykonać jako zwyczajny użytkownik (a nie administrator), dzięki czemu istnieje mniejsze prawdopodobieństwo przypadkowego zapisania zmian w oryginalnym pliku.

Przed rozpoczęciem ćwiczenia należy uruchomić X w sposób przedstawiony w rozdziale 3. Poniżej przedstawiono pełną procedurę edycji pliku *fstab-copy*:

1. Aby rozpocząć edycję pliku, trzeba otworzyć okno *xterm* i wpisać polecenie `gedit fstab-copy`. Na ekranie pojawi się okno przedstawione na rysunku 5.2. Podobnie jak w przykładzie z użyciem *vi*, zawartość tego okna będzie zależna od systemu danego użytkownika, nie ma to jednak większego znaczenia.



Rysunek 5.2. *Edytor gEdit to typowy przedstawiciel rodziny edytorów tekstu z graficznym interfejsem użytkownika. Na rysunku widoczny jest pasek menu oraz ikony wywołujące różne funkcje programu*

2. Załóżmy, że należy usunąć jeden z wpisów w tym pliku, na przykład wpis dla katalogu */windows*. Można to wykonać na wiele sposobów, które zwykle są związane z użyciem myszy. Użytkownik może kliknąć i przeciągnąć myszą lub umieścić wskaźnik na początku wiersza za pomocą myszy albo klawiszy strzałek. Przytrzymanie klawisza *Shift* i użycie strzałki w dół spowoduje zaznaczenie wiersza.

3. Aby usunąć zaznaczony wiersz, należy kliknąć ikonę *Cut* w pasku przycisków, wybrać z menu polecenie *Edit* | *Cut*, użyć skrótu *Ctrl-X* lub wcisnąć klawisz *Delete* albo *Backspace*. Po wykonaniu jednej z tych czynności wiersz */windows* zniknie z ekranu. Jeżeli w jego miejscu pojawi się pusty wiersz, należy umieścić w nim wskaźnik i wcisnąć *Backspace*.

4. Teraz możemy dodać nowy wiersz. W tym celu należy umieścić wskaźnik na początku wybranego wiersza, wpisać nowy tekst i wcisnąć klawisz *Enter*, aby oddzielić ten wiersz od następnego. Dodajmy następujący wpis:

```
/dev/da0s4      /zip    msdosfs     rw,noauto     0     0
```

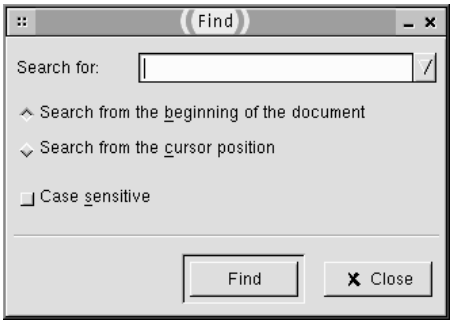
5. Wyobraźmy sobie, że konieczne jest dodanie nowego wiersza, który będzie odpowiednikiem poprzedniego wpisu, ale pozwoli FreeBSD na montowanie dysków Zip z systemem plików FFS, a nie FAT. W tym celu należy skopiować do schowka poprzedni wpis. Aby to zrobić, trzeba zaznaczyć żądany wiersz (podobnie jak w punkcie 2.) i kliknąć przycisk *Copy*, wybrać z menu polecenie *Edit* | *Copy* lub użyć klawiszy *Ctrl-C*.
6. Teraz należy umieścić wskaźnik na początku kolejnego wiersza i kliknąć przycisk *Paste*, wybrać z menu polecenie *Edit* | *Paste* lub wcisnąć klawisze *Ctrl-V*. Spowoduje to umieszczenie w wybranym miejscu dokładnej kopii wiersza z punktu 4. W razie potrzeby trzeba wcisnąć *Enter*, aby oddzielić ten wiersz od kolejnego.
7. Używając myszy lub klawiszy strzałek należy umieścić wskaźnik tuż za ciągiem */zip*.
8. Wpiszmy `-ffs`, aby przekształcić frazę */zip* w */zip-ffs*. Czterokrotne wciśnięcie klawisza *Delete* usunie spacje i wyrówna kolumny.
9. Teraz kliknijmy dwukrotnie frazę *msdosfs* w nowym wierszu, co spowoduje zaznaczenie tego ciągu.
10. Wpiszmy `ufs`, aby zastąpić ciąg *msdosfs*. Czterokrotne wciśnięcie spacji pozwoli na wyrównanie kolumn.
11. Aby zachować zmiany, należy kliknąć przycisk *Save*, wybrać z menu polecenie *File* | *Save* lub użyć klawiszy *Ctrl-S*. Jeżeli konieczne jest zapisanie pliku pod nową nazwą, trzeba użyć polecenia *File* | *Save As*. Spowoduje to otwarcie okna dialogowego, w którym należy wprowadzić nazwę nowego pliku.

Po zakończeniu edycji pliku możemy wyjść z edytora. Aby to zrobić, trzeba kliknąć przycisk *Exit*, wybrać z menu polecenie *File* | *Exit* lub użyć kombinacji klawiszy *Ctrl-Q*. Jeżeli nie zachowano jeszcze zmian w dokumencie, program „poprosi" o chęć ich zapisania.

Dodatkowe funkcje edytora gEdit

Większość opcji gEdit jest dostępnych poprzez polecenia menu, a ich zrozumienie nie powinno sprawiać problemów. Poniżej przedstawiono kilka najczęściej używanych funkcji:

- **Cofnięcie operacji**. Wybranie z menu polecenia *Edit* | *Undo* lub wciśnięcie klawiszy *Ctrl-Z* spowoduje odwołanie ostatniej operacji. Funkcja *Undo* może być użyta wielokrotnie nawet po dokonaniu dużej ilości zmian.
- **Wyszukiwanie tekstu**. Użycie polecenia *Search* | *Find* lub wciśnięcie klawisza funkcyjnego *F6* wyświetla okno dialogowe z rysunku 5.3. Szukaną frazę należy wpisać w polu *Search For* i kliknąć *Find*. W oknie dialogowym *Find* dostępnych jest kilka opcji wyszukiwania. Aby powtórzyć szukanie tej samej frazy, należy wybrać z menu polecenie *Search* | *Find Again* lub użyć skrótu *Shift-F6*.



Rysunek 5.3. *Okno dialogowe gEdit pozwala na wprowadzenie tekstu, który zostanie wyszukany w pliku tekstowym*

- **Zastępowanie tekstu**. Funkcja zamiany tekstu przypomina funkcję wyszukiwania. Użycie polecenia *Search* | *Replace* lub wciśnięcie klawisza funkcyjnego *F7* wywołuje okno dialogowe *Replace*, które jest bardzo podobne do okna *Find* z rysunku 5.3. W oknie dodano nowe pole *Replace With* oraz dwa przyciski *Replace* i *Replace All*, które pozwalają na zastąpienie jednego lub wszystkich wystąpień danej frazy przez frazę wpisaną w polu *Replace With*.
- **Ustawienia programu**. Wybranie z menu polecenia *Settings* | *Preferences* spowoduje wyświetlenie okna dialogowego *Preferences* z wieloma zakładkami, na których można zmodyfikować działanie programu. Dostępne opcje obejmują między innymi ustawienia czcionek, koloru oraz automatycznego wyrównywania tekstu.

W przeciwieństwie do edytora *vi*, gEdit pozwala na jednoczesną edycję wielu plików. Wybranie z menu polecenia *File* | *Open* lub wciśnięcie klawisza funkcyjnego *F3* powoduje wyświetlenie okna dialogowego, w którym można otworzyć wybrany plik. Po wykonaniu tej operacji pojawia się nowa zakładka z plikiem. Każdy edytowany plik jest umieszczany na oddzielnej zakładce (na rysunku 5.2 widoczna jest tylko jedna zakładka z plikiem *fstab-copy*). Aby przełączyć się pomiędzy plikami, należy kliknąć wybraną zakładkę.

Przedstawienie najważniejszych poleceń do administracji systemem

Administracja FreeBSD odbywa się zwykle poprzez edycję plików tekstowych i wprowadzanie poleceń w trybie tekstowym. Szczegółowy opis takich poleceń znajdzie się w kolejnych rozdziałach tej książki, natomiast w tej części umieszczono jedynie krótkie przedstawienie podstawowych poleceń. Można je wprowadzać zarówno w trybie tekstowym, jak i w oknie *xterm*. Większość poleceń jest dostępnych dla wszystkich użytkowników, jednak kilka z nich wymaga uprawnień administratora. Omawiane tu polecenia można ogólnie podzielić na dwie kategorie. W pierwszej grupie znajdują się polecenia dostarczające informacje o statusie lub konfiguracji systemu, natomiast druga kategoria to polecenia służące do modyfikacji ustawień systemu operacyjnego.

Warto tu również wspomnieć o możliwości łączenia poleceń. Odbywa się to zwykle za pomocą ***potokowania***, czyli funkcji przekazującej wyniki pracy pierwszego programu lub polecenia do innego programu, który traktuje je jako dane wejściowe. Potokowanie umożliwia tworzenie długiego łańcucha współpracujących narzędzi. Równie ważną rolę odgrywa ***przekierowanie***, czyli zapisanie wyników programu do pliku. Inny program może wykorzystać dane z tego pliku jako dane wejściowe. Znajomość sposobów użycia funkcji potokowania i przekierowania znacznie ułatwia wykonanie wielu zadań administracyjnych.

Polecenia dostarczające informacji o systemie

Przedstawione w tej części rozdziału polecenia zapewniają najważniejsze informacje dotyczące działania komputera i systemu operacyjnego. Dzięki tym informacjom możliwe jest między innymi rozwiązywanie problemów, planowanie sposobu instalacji oprogramowania oraz obliczanie maksymalnej liczby użytkowników, jaka może być obsłużona przez system.

Polecenie dmesg

Polecenie `dmesg` wyświetla zawartość buforu komunikatów jądra. Sposób użycia tego polecenia po uruchomieniu komputera przedstawiono w rozdziale 3. Wszystkie komunikaty generowane przez jądro są zapisywane w wewnętrznym buforze. Po załadowaniu systemu można znaleźć tam informacje dotyczące na przykład wykrytych urządzeń i błędów na niskim poziomie sprzętowym. Również w czasie pracy FreeBSD w buforze umieszczane są dodatkowe komunikaty jądra, które dotyczą między innymi problemów ze sprzętem i siecią lub niewłaściwego zachowania programów. Choć może się wydawać, że w buforze znajdują się tylko informacje dotyczące poważnych problemów, tak naprawdę większość tych komunikatów odnosi się do niegroźnych zdarzeń systemowych. Jeżeli na przykład dysk twardy jest usypiany po określonym czasie, bufor komunikatów może zostać wypełniony komunikatami dotyczącymi rozruchu tego dysku. Po zapełnieniu buforu starsze komunikaty będą sukcesywnie zastępowane nowszymi informacjami.

Dostępnych jest jedynie kilka opcji polecenia `dmesg`. Poniżej przedstawiono jego składnię:

```
dmesg [-a] [-M rdzeń] [-N system]
```

Opcja `-a` powoduje wyświetlenie rozszerzonego raportu, który obejmuje informacje pochodzące z konsoli i dziennika systemowego. Opcja `-M` *`rdzeń`* nakazuje analizę ***rdzenia jądra***, czyli pliku generowanego w przypadku wystąpienia niektórych poważnych awarii. W większości przypadków użycie tej opcji nie jest potrzebne. Opcja `-N` *`system`* pozwala na analizę buforu komunikatów jądra innego systemu.

Polecenie df

Polecenie `df` informuje o dostępnej przestrzeni dyskowej. Te dane będą bardzo przydatne na przykład w czasie planowania instalacji nowego oprogramowania, szczególnie jeśli partycje są już w dużym stopniu zapełnione. Oto pełna składnia polecenia `df`:

```
df [-b | -h | -H | -k | -m | -P] [-ailn] [-t typ] [plik | system_plików ...]
```

Znaczenie poszczególnych opcji przedstawiono poniżej:

- **Opcje formatowania**. Pierwsza grupa opcji decyduje o jednostkach użytych w raportach dotyczących przestrzeni dyskowej. Opcja `-b` oznacza bloki o wielkości 512 bajtów, opcje `-h` i `-H` nakazują użycie kilobajtów, megabajtów itd. (opartych odpowiednio na potęgach 10 i 2). Opcja `-k` oznacza kilobajty (1 kilobajt to 1024 bajtów), natomiast opcja `-m` oznacza megabajty (1 megabajt to 1 048 576 bajtów). Użycie `-P` nakazuje wyświetlenie wyników w formacie zgodnym ze standardem POSIX.

Użytkownicy komputerów zwykle podają wielkość dysków w jednostkach opartych na potęgach 2. Jeden kilobajt to 1024 bajtów, a jeden megabajt ma 1 048 576 bajtów. Producenci dysków twardych używają jednostek opartych na potęgach 10, gdzie 1 kilobajt ma 1000 bajtów, a jeden megabajt to 1 000 000 bajtów.

- `-a` — ta opcja powoduje wyświetlenie informacji o wszystkich punktach montowania, włącznie z systemami plików montowanymi w taki sposób, że w normalnej sytuacji zostałyby zignorowane przez polecenie `df`.
- `-i` — dzięki tej opcji polecenie `df` wyświetli informacje o dostępnych ***i-węzłach*** (***inodes***), czyli strukturach danych używanych wewnętrznie do adresowania plików. Jeżeli na dysku zabraknie i-węzłów, nie będzie możliwe zapisanie dodatkowych plików, nawet gdy nie brakuje wolnego miejsca. Może to stać się problemem dla użytkowników zapisujących na dysku dużą liczbę małych plików.
- `-l` — ta opcja ogranicza działanie polecenia `df` tylko do lokalnych systemów plików (na przykład dysk twardy lub napęd CD-ROM). Informacje o sieciowych systemach plików nie zostaną wyświetlone.
- `-n` — ta opcja powoduje wyświetlenie informacji już znajdujących się w pamięci. W normalnej sytuacji polecenie `df` nakazuje FreeBSD zbadanie wszystkich systemów plików i uzyskanie informacji o ich bieżącym stanie. Jeżeli jednak sieć nie będzie dostępna, polecenie `df` nie będzie mogło zebrać danych dotyczących sieciowych systemów plików.
- `-t` *typ* — dzięki tej opcji możliwe jest wyświetlenie informacji dotyczących systemów plików określonego typu. Aby wybrać kilka typów, należy je rozdzielić przecinkami; na przykład polecenie `df -t ufs,cd9660` przedstawi dane dotyczące tylko systemów plików FFS i ISO-9660.
- *plik* | *system_plików* — aby ograniczyć działanie polecenia `df` do określonych systemów plików, należy podać nazwy wybranych urządzeń (na przykład */dev/ad0s2h*) lub ich punkty montowania, jak na przykład */home*.

Informacje zapewniane przez polecenie `df` obejmują nazwę urządzenia, wielkość systemu plików, wykorzystane miejsce, dostępne miejsce, procent wykorzystania oraz punkt montowania systemu plików. Przedstawione poniżej polecenie wyświetla dane dotyczące jednego systemu; zostały one sformatowane w celu łatwiejszej analizy:

```
$ df -h
Filesystem        Size     Used    Avail Capacity  Mounted on
/dev/ad0s2a       124M      62M      52M    54%    /
```

```
devfs              1.0K     1.0K       0B   100%    /dev
/dev/ad0s2h        1.8G     486M      1.2G    29%    /home
/dev/ad0s2f        145M      31M      103M    23%    /tmp
/dev/ad0s2g        1.6G     1.2G      269M    82%    /usr
/dev/ad0s2e        145M     7.4M      126M     6%    /var
/dev/ad0s1         2.0G     697M      1.3G    34%    /windows
procfs             4.0K     4.0K       0B   100%    /proc
speaker:/home      4.5G     1.7G      2.8G    38%    /speaker/home
```

Opierając się na powyższych wynikach, możemy stwierdzić, że główny system plików (*/*) ma wielkość 124 MB i jest zapełniony w 54%, natomiast katalog */home* jest zapełniony w 29%. System plików */speaker/home* został zamontowany poprzez sieć, o czym świadczy nazwa urządzenia *speaker:/home* (jest to specyfikacja NFS). Systemy plików */dev* i */proc* to specjalne wirtualne systemy plików, które nie są powiązane z rzeczywistymi partycjami lub urządzeniami, ale są tworzone w razie potrzeby. Z tego powodu oba te systemy zgłaszają brak wolnego miejsca. Znacznie groźniejsza jest informacja o wykorzystaniu partycji */usr* w 82%. Może to spowodować problemy w przypadku próby instalacji programów, których wielkość przekracza 269 MB. Zaleca się, aby na partycji */usr* (lub na partycji głównej, jeżeli nie ma oddzielnej partycji */usr*) zawsze zapewnić co najmniej kilkanaście megabajtów wolnego miejsca.

Polecenie man

Polecenie `man` stanowi klucz do dokumentacji systemowej FreeBSD. W najprostszej formie ma ono postać `man` *`temat`*, gdzie temat może być poleceniem, nazwą pliku, wywołaniem systemowym lub nazwą opcji. Działanie `man` można zmodyfikować na wiele sposobów, na przykład opcja `-P` *`pager`* spowoduje użycie wybranego pagera do wyświetlenia informacji. Aby użyć pagera *less*, a nie domyślnego pagera *more*, należy wpisać polecenie `man -P less` *`temat`*. Zmiana domyślnego pagera dla polecenia `man` odbywa się poprzez ustawienie zmiennej środowiskowej *PAGER* w pliku konfiguracyjnym powłoki.

Polecenie uptime

Polecenie `uptime` nie ma żadnych argumentów; jego użycie spowoduje wyświetlenie czasu, jaki upłynął od momentu uruchomienia komputera, liczby zalogowanych użytkowników oraz średniego obciążenia procesora. Poniżej przedstawiono sposób użycia polecenia `uptime`:

```
$ uptime
1:49AM  up 5 days, 5 mins, 5 users, load averages: 0.00, 0.00, 0.00
```

Liczba użytkowników tak naprawdę oznacza liczbę zalogowanych osób lub uruchomionych sesji *xterm*. W powyższym przykładzie jeden użytkownik wykorzystywał pięć sesji *xterm*. Na podstawie wyników polecenia `uptime` nie można stwierdzić, z jakim typem użytkownika mamy do czynienia.

Trzy liczby zgłaszane jako *load averages* informują o zapotrzebowaniu na czas procesora w ciągu trzech okresów (ostatnia minuta, 5 minut i 15 minut). Wartość *0.00* wskazuje, że procesor nie jest praktycznie obciążony, natomiast wartość *1.00* oznacza, że procesor pracował z pełną wydajnością (na przykład w przypadku uruchomienia programu

wykonującego skomplikowane obliczenia lub otwarcia wielu narzędzi wymagających pełnej mocy procesora). Jeżeli przeciętne obciążenie przekracza *1.00*, system nie będzie mógł zapewnić wystarczającej mocy obliczeniowej procesora dla wszystkich zadań. Choć poszczególne aplikacje będą działały poprawnie, ich wydajność nie będzie zbyt wysoka. Należy zauważyć, że wartość *load averages* nie mówi nic o prędkość procesora. Program do analizy danych może spowodować pełne obciążenie zarówno procesora Pentium 100 MHz, jak i Pentium 4 o prędkości 2 GHz.

Polecenie ps

Polecenie `ps` wyświetla informacje o działających procesach. Liczba dostępnych opcji `ps` jest tak duża, że ich szczegółowe omówienie wymagałoby kilkunastu stron książki. Aby uzyskać więcej informacji na ten temat, powinniśmy zapoznać się ze stroną dokumentacji systemowej poświęconą temu poleceniu. Poniżej przedstawiono najczęściej używane opcje:

- `-a` — wyświetla informacje dotyczące procesów wszystkich użytkowników. Domyślnie dostępne są dane dotyczące jedynie procesów osoby, która użyła polecenia `ps`.
- `-h` — powtarza nagłówki identyfikujące poszczególne kolumny wyników, dzięki czemu będą one widoczne na każdej stronie z wynikami.
- `-j` — wyświetla dodatkowe informacje o każdym procesie.
- `-l` — wyświetla dodatkowe informacje o każdym procesie.
- `-m` — sortuje wyniki według wykorzystania pamięci.
- `-p` *PID* — wyświetla informacje dotyczące procesu o podanym identyfikatorze PID.
- `-r` — sortuje wyniki według obciążenia procesora.
- `-U` *nazwa_użytkownika* — wyświetla informacje o procesach należących do konkretnego użytkownika.
- `-u` — wyświetla dodatkowe informacje o każdym procesie.
- `-v` — wyświetla dodatkowe informacje o każdym procesie.
- `-w` — powoduje wyświetlenie wyników w formacie 132-kolumnowym, a nie w domyślnym formacie 80-kolumnowym. Ta opcja będzie szczególnie przydatna dla użytkowników pracujących na dużych monitorach. Użycie opcji `-w` pozwala także na przekierowanie wyników polecenia `ps` do pliku, który zostanie przetworzony przez inny proces lub edytor tekstu.
- `-x` — wyświetla informacje o procesach, które nie są powiązane z terminalem (na przykład o demonach).

Jedną z najważniejszych informacji zapewnianych przez polecenie `ps` jest identyfikator PID każdego procesu. Ten numer jest często wykorzystywany jako parametr innych poleceń, takich jak `renice` i `kill`; zostaną one przedstawione w podrozdziałach „Polecenia nice i renice” oraz „Polecenia kill i killall”. W wielu przypadkach polecenie `ps` jest używane tylko po to, aby uzyskać identyfikator PID konkretnego procesu.

Dodatkowe informacje wyświetlane przez niektóre opcje (takie jak `-j`, `-l`, `-u` i `-v`) obejmują identyfikator procesu macierzystego (PPID), godzinę i datę uruchomienia procesu oraz wykorzystanie pamięci (podawane w różnych formatach).

Jednym z najbardziej przydatnych wariantów tego polecenia jest `ps -aux`. To polecenie wyświetla informacje o wszystkich działających procesach wraz z nazwami powiązanych użytkowników, obciążeniem procesora i pamięci (w procentach) oraz poleceniami użytymi do uruchomienia każdego z tych procesów. Bardzo często wyniki powyższego polecenia są potokowane (zobacz podrozdział „Potoki") poprzez narzędzie *grep*, które wyszukuje określony ciąg w danych wejściowych. Pozwala to odnaleźć poszczególne egzemplarze uruchomionego programu. Aby na przykład odnaleźć wszystkie egzemplarze pagera *more*, należy użyć poniższego polecenia:

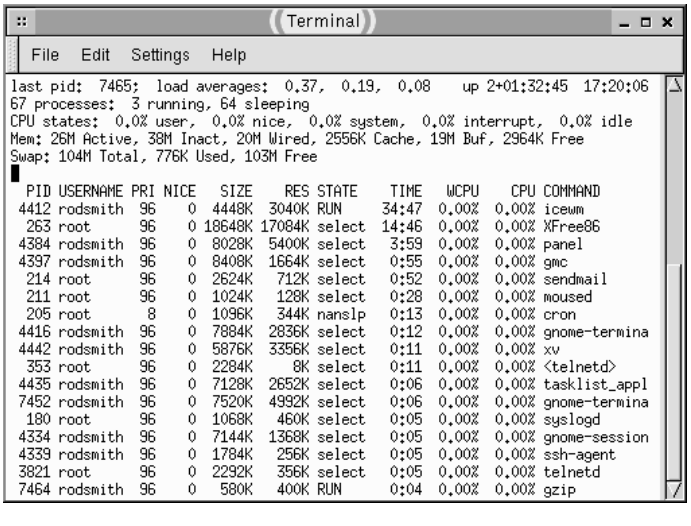
```
$ ps -aux | grep more
rodsmith  7445  0.0  0.2   344   55  p3 R+    5:12PM   0:00.02 grep more
rodsmith  7384  0.0  1.1  1372  245  p2 I+    4:57PM   0:00.08 more
```

Powyższe polecenie często przechwytuje nie tylko proces docelowy (w tym przypadku *more*), ale także proces *grep*.

Polecenie top

W pewnym sensie polecenie `top` stanowi wariant polecenia `ps`, gdyż wyświetla listę wszystkich uruchomionych programów. Ta lista jest jednak sortowana dynamicznie według obciążenia procesora, dzięki czemu `top` jest wyjątkowo przydatnym narzędziem pozwalającym na odnalezienie błędnych procesów, które działają w pętli i powodują niepotrzebne wykorzystanie mocy procesora. Rysunek 5.4 przedstawia działanie polecenia `top`.

Rysunek 5.4. *Polecenie top aktualizuje wyniki w czasie rzeczywistym, aby pokazać zmiany zachodzące na liście aktywnych procesów oraz w systemie*



Polecenie `top` ma wiele wariantów; na przykład w środowisku GNOME dostępne jest narzędzie `gtop` z graficznym interfejsem użytkownika. Choć takie programy zapewniają identyczne wyniki jak polecenie `top` w trybie tekstowym, mogą być bardziej intuicyjne dla osób preferujących graficzne środowisko pracy. Warto wiedzieć, że narzędzia tego

typu mogą być użyte nawet w innym środowisku; na przykład `gtop` może być uruchomiony w KDE lub w dowolnym innym menedżerze okien (oczywiście pod warunkiem że uruchomiono serwer X i zapewniono odpowiednie biblioteki GNOME).

Polecenia do manipulacji systemem

Uzyskanie informacji dotyczących systemu pozwoli przygotować diagnozę problemu lub zaplanować zmiany. Potrzebna jest jednak metoda pozwalająca na naprawienie systemu lub implementację niezbędnych zmian. W dalszej części książki można znaleźć opis niezbędnych procedur, natomiast w tej części rozdziału przedstawię najczęściej używane polecenia służące do manipulacji systemem.

Polecenia nice i renice

Poszczególne programy zgłaszają systemowi zapotrzebowanie na moc obliczeniową procesora. Jednym z zadań jądra FreeBSD jest obsługa tych żądań oraz mediacja w przypadku powstania konfliktu. Wyobraźmy sobie, że użytkownicy *ajones* i *bsmith* jednocześnie uruchamiają programy wykonujące skomplikowane obliczenia. W jaki sposób zostanie podzielony czas procesora? W normalnej sytuacji system przyzna obu użytkownikom identyczną moc obliczeniową, dzięki czemu żadne zadanie nie będzie uprzywilejowane. Czasami jednak konieczna jest zmiana domyślnego zachowania FreeBSD. Aby zmienić priorytet uruchamianego procesu, należy użyć polecenia `nice` w połączeniu z kodem priorytetu. Powiązane polecenie `renice` umożliwia zmianę priorytetu już uruchomionego zadania.

Poniżej przedstawiono składnię polecenia `nice`:

```
nice [-liczba] polecenie [argumenty]
```

`polecenie` to nazwa polecenia lub programu, jaki zostanie uruchomiony, a *`argumenty`* to po prostu jego opcje. Liczba to kod priorytetu. Domyślnie każdy proces otrzymuje priorytet 0. Wyższa *`liczba`* (maksymalnie 20) zmniejsza priorytet procesu, natomiast mniejsza *`liczba`* (aż do –20) zwiększa priorytet zadania. Kod priorytetu należy poprzedzić myślnikiem (–). Aby zmniejszyć priorytet procesu, należy więc użyć liczby ujemnej, na przykład:

```
$ nice -15 numbercrunch
```

Aby zwiększyć priorytet, należy wpisać myślnik i liczbę ujemną, na przykład:

```
# nice --15 numbercrunch
```

Proszę zauważyć, że w powyższych przykładach użyto innych znaków zachęty. Jest to spowodowane faktem, że dowolny użytkownik może zmniejszyć priorytet procesu, ale tylko użytkownik *root* ma prawo do zwiększenia uprzywilejowania zadania. Domyślną wartością polecenia `nice` jest 10.

Polecenie `nice` jest używane dość rzadko, choć może być przydatne w przypadku procesów o niskim priorytecie, które wymagają jednak dużej mocy obliczeniowej. Przykładem

jest program SETI@Home (*http://setiathome.ssl.berkeley.edu*), który powoduje znaczne obciążenie procesora. Uruchomienie tego programu z niską wartością `nice`, na przykład `20`, umożliwi bardziej wydajną pracę innych aplikacji.

Narzędzie `renice` uzupełnia funkcje `nice` o możliwość zmiany priorytetu już uruchomionego procesu. Poniżej przedstawiono składnię polecenia `renice`:

```
renice priorytet [[-p] pid...] [[-g] pgrp...] [[-u użytkownik...]
```

Procesy są identyfikowane poprzez ich identyfikator PID (można go sprawdzić poprzez użycie polecenia `ps`, `top` lub podobnych narzędzi), nazwę grupy właściciela lub nazwy właścicieli. Poniższe polecenie pozwoli zwiększyć priorytet wszystkich procesów należących do użytkownika *ajones*:

```
# renice -1 ajones
```

Proszę zauważyć, że w przeciwieństwie do polecenia `nice`, polecenie `renice` nie wymaga użycia myślnika przed wartością priorytet. Oznacza to, że użycie liczby ujemnej spowoduje zwiększenie priorytetu. W razie problemu z rozróżnieniem identyfikatorów PID, nazw grup lub nazw użytkowników, należy dołączyć opcje `-p`, `-g` oraz `-u`. Tylko użytkownik *root* może zwiększyć priorytet wybranego procesu. Użytkownicy nie będący administratorami mogą stosować polecenie `renice` tylko dla własnych procesów.

Jeżeli zwyczajny użytkownik wykorzystał polecenie `nice` lub `renice` do zmniejszenia priorytetu procesu, nie będzie mógł zwiększyć tego priorytetu ani przywrócić go do stanu początkowego.

Jeżeli po uruchomieniu niektórych programów ogólna wydajność systemu spada, polecenie `renice` pozwoli przywrócić jego prawidłową pracę. Największe różnice można zaobserwować w czasie pracy aplikacji intensywnie korzystających z procesora, takich jak programy służące do symulacji naukowych lub edycji grafiki.

Polecenia kill i killall

Polecenia `nice` i `renice` pozwalają przydzielić moc obliczeniową procesora dla najważniejszych procesów uruchomionych na komputerze. Często jednak to nie wystarcza, gdyż procesy zawieszają się i pracują w pętli, co powoduje marnowanie zasobów systemowych. Zawieszenie programu może spowodować również problemy z działaniem innych aplikacji, a usunięcie błędnego procesu z pamięci w prosty sposób jest czasami niemożliwe. Na szczęście, z pomocą przychodzą polecenia `kill` i `killall` pomagające przerwać działanie wszystkich procesów, włącznie z programami, które nie reagują na żadne czynności wykonywane przez użytkownika.

Poniżej przedstawiono składnię polecenia `kill`:

```
kill [-l] [[-s] nazwa_sygnału | numer_sygnału] PID...
```

Polecenie `kill` w najprostszej formie nie wymaga żadnych parametrów poza identyfikatorem PID, który można uzyskać przy użyciu poleceń `ps` lub `top`. Jeżeli chcemy odnaleźć i unicestwić program o nazwie *xv*, musimy użyć następujących poleceń:

```
# ps ax | grep xv
  7672  p2  RN+    0:00.02 grep xv
  4442  p5  I      0:13.40 xv
# kill 4442
```

Polecenie `kill` unicestwia proces, wysyłając do niego ***sygnał*** — specjalny kod używany do komunikacji między procesami. Domyślnie wysyłany jest sygnał o numerze `15`, znany także jako `TERM`. Większość procesów odpowiada na ten sygnał, przerywając pracę w kontrolowany sposób. Niestety, niektóre zawieszone procesy nie reagują na sygnał `TERM`. Aby pozbyć się takich procesów, należy użyć sygnału o numerze `9`, czyli `KILL`; na przykład polecenie `kill -s 9 4442` wyśle sygnał `KILL` do procesu o numerze `4442`. Użycie tego sygnału nakazuje natychmiastowe unicestwienie procesu przez jądro bez możliwości zamknięcia otwartych plików lub wykonania innych procedur końcowych. W tabeli 5.1 przedstawiono pozostałe sygnały i ich znaczenie. Aby uzyskać kompletną listę sygnałów, należy wpisać polecenie `kill -l`.

Tabela 5.1. *Typowe sygnały i ich znaczenie*

Nazwa sygnału	Numer sygnału	Znaczenie
`HUP`	1	Rozłączenie
`INT`	2	Przerwanie
`QUIT`	3	Wyjście
`ABRT`	6	Anulowanie
`KILL`	9	Nieodwołalne unicestwienie
`ALRM`	14	Alarm
`TERM`	15	Zamknięcie procesu

Polecenie `killall` wykonuje podobną funkcję jak `kill`, ale pozwala na unicestwienie wszystkich procesów zgodnych z podanym wzorcem. Aby zamknąć program *xv* z wcześniejszego przykładu, wystarczy użyć polecenia `killall xv`. Należy jednak pamiętać, że polecenie `killall` może spowodować nieoczekiwane zamknięcie innych procesów. Jeżeli program *xv* został uruchomiony przez dwie osoby, administrator może łatwo spowodować unicestwienie obu egzemplarzy *xv*.

Polecenie `killall` w niektórych wersjach Uniksa ma inne działanie niż we FreeBSD. Użycie tego polecenia może spowodować zamknięcie wszystkich procesów danego użytkownika, z wyjątkiem rodziców samego procesu `killall`. Przed wpisaniem tego polecenia należy zapoznać się z dokumentacją systemową.

Polecenia `kill` i `killall` mogą być używane zarówno przez administratorów, jak i przez zwyczajnych użytkowników. Oczywiście posiadacz normalnego konta może przerwać działanie tylko własnych procesów.

Przekierowanie

Funkcja przekierowania pojawiała się już kilkakrotnie we wcześniejszych rozdziałach tej książki. To narzędzie znacznie ułatwia administrację FreeBSD w trybie tekstowym,

jednak może być przydatne również dla zwyczajnych użytkowników. Dzięki tej funkcji możliwe jest przekierowanie danych wejściowych lub wyników programu z lub do pliku, co stanowi alternatywę dla klawiatury lub monitora.

Aby zrozumieć działanie funkcji przekierowania, należy wiedzieć, że większość programów pracujących w trybie tekstowym może odbierać dane i wysyłać wyniki do wielu źródeł (są one często nazywane ***strumieniami***). W większości przypadków wszystkie strumienie wejściowe są powiązane z klawiaturą, z której korzysta użytkownik, a wszystkie strumienie wyjściowe są powiązane z ekranem, oknem *xterm* lub zdalnym procesem logowania użytkownika. Domyślny strumień wejściowy nosi nazwę ***wejścia standardowego*** (lub *stdin*), a strumień wyjściowy jest nazywany ***wyjściem standardowym*** (lub *stdout*). Istnieje jeszcze jeden ważny strumień wyjściowy, który nosi nazwę ***błędu standardowego*** lub *stderr*. Ten dodatkowy strumień służy do obsługi komunikatów o błędach. Strumień *stderr* jest zwykle powiązany z tą samą konsolą co *stdout*, ale możliwe jest przekierowanie go w inny sposób.

Przekierowanie oznacza powiązanie tych strumieni wejściowych i wyjściowych z innym źródłem lub miejscem docelowym. Pozwala to na przykład na przesłanie wyników pracy programu bezpośrednio do pliku, a nie na ekran. Otrzymany plik wynikowy można otworzyć w wybranym edytorze tekstu lub przeszukać przy użyciu odpowiednich narzędzi. Aby przekierować *stdout* do pliku, należy użyć symbolu > oraz podać nazwę pliku docelowego, w którym zostaną zapisane wyniki. Poniższe polecenie pozwoli zapisać długi listing w pliku tekstowym:

```
$ ls -l > listing.txt
```

Po wykonaniu powyższego polecenia możemy użyć pagera (na przykład `less`) do przejrzenia wyników, załadować plik do edytora tekstu lub wykonać inną operację na pliku wynikowym. Niektóre programy korzystają ze strumienia *stderr* częściej niż ze *stdout*, ponieważ użytkownik powinien natychmiast otrzymać informację o powstałych problemach. Czasami jednak konieczne jest zapisanie do pliku komunikatów o błędach. Aby to zrobić, należy użyć przekierowania *stdout*, w którym symbol > zostanie zastąpiony przez 2>. Jeżeli program o nazwie *bigerr* intensywnie wykorzystuje strumień *stderr*, jego przekierowanie możemy wykonać w następujący sposób:

```
$ bigerr 2> komunikaty_bledow_bigerr.txt
```

Aby jednocześnie przekierować strumienie *stdout* i *stderr*, możemy użyć poniższego polecenia:

```
$ bigerr > wyniki_bigerr.txt 2> komunikaty_bledow_bigerr.txt
```

Wszystkie te polecenia powodują nadpisanie istniejącego pliku nowym plikiem tekstowym. Aby dodać wyniki pracy programu do pliku, należy użyć dwóch znaków >, na przykład:

```
$ ls -l >> listing.txt
```

Wiele programów pozwala na wprowadzenie danych z klawiatury. Jeżeli aplikacja została napisana w odpowiedni sposób (na przykład jest to skrypt do wprowadzania imion,

nazwisk, adresów i numerów telefonów), możliwe będzie użycie przekierowania wejścia, aby odczytać dane wejściowe z pliku. Do tego celu używany jest znak <. Poniższe polecenie powoduje przekazanie pliku *adres.txt* do skryptu o nazwie *getinfo*:

```
$ getinfo < adres.txt
```

Możliwe jest połączenie funkcji przekierowania wejścia i wyjścia w jednym poleceniu, na przykład:

```
$ getinfo < adres.txt > wyniki.txt
```

Powyższe polecenie przekazuje zawartość pliku *adres.txt* jako dane wejściowe, a następnie wysyła wyniki programu do pliku *wyniki.txt*. Ta forma przekierowania jest często używana w skryptach działających automatycznie; zostaną one opisane w rozdziale 31.

Potoki

Potoki są blisko powiązane z funkcją przekierowania. Dzięki potokom możliwe jest powiązanie dwóch programów w taki sposób, że wyniki pierwszego programu są przekazywane jako dane wejściowe do drugiego programu. W jednym potoku można znaleźć się większa liczba programów, co pozwala na powiązanie wielu prostych narzędzi w złożoną strukturę, która wykonuje skomplikowane zadania.

Potoki pojawiły się już w przykładach z użyciem poleceń `ps` i `grep` (zobacz podrozdziały „Polecenie ps" i „Polecenia kill i killall"). Pierwsze z tych poleceń wyświetla zwykle bardzo rozbudowane wyniki; każdy działający proces jest przedstawiany w oddzielnym wierszu, co nawet w przypadku prostego systemu FreeBSD oznacza dużą liczbę danych. Aby uzyskać informacje tylko o kilku wybranych procesach, należy użyć funkcji potokowania do narzędzia *grep*. Program *grep* wyszukuje wiersze, które zawierają żądaną frazę, dzięki czemu możliwe jest jego użycie do przefiltrowania wyników pracy polecenia `ps`. Jako operator potoku stosowany jest symbol pionowej kreski (`|`). Poniżej przedstawiono przykład użycia potoku i narzędzia *grep*:

```
$ ps -aux | grep vx
```

Powyższe polecenie wyszukuje i wyświetla na ekranie wszystkie wiersze wynikowe polecenia `ps`, które zawierają ciąg *vx*. Ta technika znacznie ułatwia analizę rozbudowanych informacji. Oczywiście potoki można wykorzystać do wielu innych zastosowań, takich jak przesłanie plików odnalezionych przez `find` do programu służącego do archiwizacji danych. Możliwe jest również powiązanie wielu narzędzi do przetwarzania dźwięku w celu utworzenia zadziwiających efektów dźwiękowych. Poniższe polecenie pozwala na przesłanie wyników polecenia `ps` do wybranego pagera (*less* lub *more*):

```
$ ps -aux | less
```

Więcej informacji na temat poleceń *grep* i *find* można znaleźć w rozdziale 8.

Narzędzia administracyjne z graficznym interfejsem użytkownika

W tym rozdziale skoncentrowaliśmy się na narzędziach administracyjnych, które pracują w trybie tekstowym. Nie należy jednak zapominać, że możliwe jest zarządzanie FreeBSD przy użyciu programów z graficznym interfejsem użytkownika. Większość administratorów FreeBSD wybiera jednak narzędzia tekstowe, co jest spowodowane przyzwyczajeniem użytkowników Uniksa, a także faktem, że w domyślnej konfiguracji systemu nie są dostępne narzędzia z interfejsem GUI (z pewnym wyjątkiem, który zostanie niedługo opisany). Takie programy mogą być jednak użyte w niektórych sytuacjach, na przykład kiedy administrator nie zna dobrze konkretnego obszaru systemu, a konieczne jest szybkie uruchomienie tego podsystemu. Również mniej doświadczeni użytkownicy mogą wykonać proste zadania administracyjne z użyciem narzędzi z interfejsem GUI. Należy jednak pamiętać, że zwykle są one bardziej ograniczone niż programy pracujące w trybie tekstowym. Dlatego zalecam poznanie standardowych narzędzi systemowych, które zapewniają potężniejsze funkcje i pełną kontrolę nad FreeBSD.

Brakuje tu niestety miejsca na zamieszczenie pełnej listy narzędzi administracyjnych z graficznym interfejsem użytkownika, a nawet na bardziej szczegółowe omówienie najpopularniejszych programów. Z tego powodu przedstawiono jedynie kilka podstawowych narzędzi wraz z krótkim opisem ich funkcji:

- **sysinstall** — nie jest to typowe narzędzie z graficznym interfejsem użytkownika, ale stanowi standardowy element FreeBSD i zapewnia niektóre funkcje, które są dostępne w narzędziach graficznych. *sysinstall* steruje procesem instalacji FreeBSD, co opisano w rozdziale 2. Program można uruchomić również po zakończeniu instalacji zarówno w trybie tekstowym, jak i w oknie *xterm*. Aby to zrobić, należy wpisać polecenie `sysinstall`. Pozwoli to na instalację dodatkowego oprogramowania, zmianę ustawień X, ustawienie strefy czasowej i wykonanie innych zadań administracyjnych. Opis poszczególnych funkcji tego narzędzia można znaleźć w wielu rozdziałach tej książki.
- **Webmin** — to narzędzie jest dostępne pod adresem *http://www.webmin.net*; służy ono do administracji FreeBSD oraz innymi uniksowymi systemami operacyjnymi poprzez przeglądarkę internetową. Webmin zapewnia ogromną liczbę funkcje, które są dostępne poprzez moduły administracyjne obsługujące konkretne zadania lub programy serwerowe.
- **SWAT** — program ***Samba Web Administration Tool*** (***SWAT***) jest instalowany wraz z Sambą, która zostanie opisana w rozdziale 18. Podobnie jak Webmin, SWAT wykorzystuje przeglądarkę internetową. Ponieważ program zapewnia dostęp do praktycznie wszystkich funkcji Samby, możliwa jest szybka konfiguracja nawet skomplikowanych ustawień tego podsystemu.
- **CUPS** — ***Common Unix Printing System*** (***CUPS***) stanowi alternatywę dla standardowych narzędzi do obsługi drukarek we FreeBSD (zobacz rozdział 9.). CUPS to kolejne narzędzie wykorzystujące przeglądarkę internetową do konfiguracji ustawień. Więcej informacji na temat systemu zarządzania drukarkami CUPS można znaleźć na stronie *http://www.cups.org*.

- **Inne narzędzia** — niektóre inne programy posiadają własne narzędzia administracyjne z graficznym interfejsem użytkownika lub umożliwiają konfigurację poprzez przeglądarkę internetową. Dotyczy to głównie rzadziej używanych programów, takich jak *fetchmail* (*http://www.tuxedo.org/~esr/fetchmail*) i Leafnode (*http://www.leafnode.org*; na stronie *http://leafwa.sourceforge.net* można znaleźć moduł administracyjny Leafwa z interfejsem GUI.

Coraz większą popularnością cieszą się narzędzia administracyjne wykorzystujące przeglądarkę internetową, które pozwalają zarządzać systemem FreeBSD z innego komputera w sieci lokalnej, niezależnie od zainstalowanego systemu operacyjnego. Użycie takich narzędzi poza siecią lokalną jest potencjalnym zagrożeniem dla bezpieczeństwa systemu ze względu na ryzyko przechwycenia hasła administratora i przejęcia kontroli nad serwerem. Podobne ryzyko istnieje również w sieci lokalnej, która nie jest w pełni bezpieczna. Administrator powinien używać takie programy z rozwagą.

Podsumowanie

Administracja FreeBSD odbywa się poprzez edycję plików tekstowych zawierających domyślne ustawienia systemu oraz przy użyciu specjalnych narzędzi, które wykonują określone zadania. Oznacza to konieczność poznania sposobu ich działania i wpływu na system operacyjny. Równie ważna jest umiejętność edycji plików tekstowych w jednym z wielu dostępnych edytorów tekstu.

Ten rozdział stanowi jedynie wprowadzenie do administracji systemem FreeBSD. Choć przedstawiono tu kilka istotnych narzędzi i plików konfiguracyjnych, najważniejsze metody zarządzania systemem zostaną opisane w kolejnych rozdziałach.