helion.pl
KSIĘGARNIA INTERNETOWA · WYDAWNICTWO

IDŹ DO
PRZYKŁADOWY ROZDZIAŁ
SPIS TREŚCI

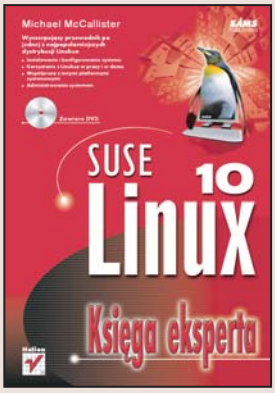
KATALOG KSIĄŻEK
KATALOG ONLINE
ZAMÓW DRUKOWANY KATALOG

TWÓJ KOSZYK
DODAJ DO KOSZYKA

CENNIK I INFORMACJE
ZAMÓW INFORMACJE O NOWOŚCIACH
ZAMÓW CENNIK

CZYTELNIA
FRAGMENTY KSIĄŻEK ONLINE

SUSE Linux 10. Księga eksperta



Autor: Michael McCallister
Tłumaczenie: Adam Jarczyk
ISBN: 83-246-0409-X
Tytuł oryginału: SUSE Linux 10 Unleashed
Format: B5, stron: 688

Wyczerpujący przewodnik po jednej z najpopularniejszych dystrybucji Linuksa

- Instalowanie i konfigurowanie systemu
- Korzystanie z Linuksa w pracy i w domu
- Współpraca z innymi platformami systemowymi
- Administrowanie systemem

SUSE Linux to jedna z najbardziej znanych kompletnych dystrybucji Linuksa. Jej popularność w Europie i Ameryce wzrasta w ogromnym tempie, między innymi dzięki temu, że rozwijaniem systemu zajęła się firma Novell. Wersja 10 powstała w ramach projektu Open SUSE, w którym brały udział tysiące użytkowników i programistów z całego świata. Efektem jest stabilny, wydajny i niezawodny system operacyjny, świetnie sprawdzający się jako podstawa działania zarówno stacji roboczych i komputerów domowych, jak i serwerów sieciowych. Ogromną zaletą dystrybucji SUSE jest maksymalnie uproszczony proces instalacji oraz imponująca ilość narzędzi i aplikacji dołączanych do systemu.

„SUSE Linux 10. Księga eksperta" to doskonałe źródło wiedzy na temat najnowszej wersji tej dystrybucji. Zawiera informacje, dzięki którym poznasz system i staniesz się jego świadomym użytkownikiem. Nauczysz się go instalować, konfigurować oraz przeprowadzać operacje administracyjne. Dowiesz się, jak korzystać z dołączonych do Linuksa aplikacji oraz jak instalować nowe narzędzia z pakietów dystrybucyjnych. Poznasz tajniki uruchamiania serwerów sieciowych w oparciu o SUSE Linux i sposoby przygotowywania go do współpracy z innymi systemami operacyjnymi.

W książce tej znajdziesz omówienie następujących zagadnień:

- instalacja z płyty DVD,
- praca z powłoką tekstową,
- środowiska KDE i Gnome,
- konfiguracja drukarki,
- korzystanie z pakietu OpenOffice,
- współpraca z Windows i Mac OS,
- korzystanie z internetu,
- zarządzanie systemem plików i kontami użytkowników,
- konfiguracja serwerów sieciowych,
- programowanie w Linuksie.

Poznaj i wykorzystaj nowoczesną dystrybucję Linuksa!

Wydawnictwo Helion
ul. Kościuszki 1c
44-100 Gliwice
tel. 032 230 98 63
e-mail: helion@helion.pl

Spis treści

O autorze 19

Wprowadzenie 21

Część I Instalacja i konfiguracja 27

Rozdział 1. Witamy w systemie SUSE Linux 29

Czym jest SUSE Linux? 29
SUSE Linux 30
OpenSUSE.org 32
Program instalacyjny SUSE Linux 33
Systemy plików w SUSE Linux 34
64-bitowy SUSE Linux 34
Pomoc: dokumentacja i inne źródła 35
Dokumentacja drukowana i SUSE Help Center 35
Przeszukiwanie Help Center 35
Strony man i info 36
Pomoc techniczna online 37
Linux Documentation Project 38
Listy dyskusyjne SUSE 38
Grupy użytkowników Linuksa 40
Odnośniki 40

Rozdział 2. Przygotowania do instalacji 43

Planowanie instalacji SUSE Linux 44
Wymogi sprzętowe 45
Linux w laptopach 48
SUSE Linux i przestarzały sprzęt 48
Unikanie potencjalnych problemów ze sprzętem 49
Modemy programowe 49
Urządzenia USB 50
Urządzenia „peryferyjne" na płycie głównej 51
Dwa systemy, podział na partycje i inne problemy 51
Współistnienie z systemem Windows 52
Jak podzielić dyski na partycje? 53
Odnośniki 54

Rozdział 3. Instalacja SUSE Linux 55
Metody instalacji systemu 55
Przygotowania do instalacji z płyty CD-ROM lub DVD 55
Instalacja SUSE Linux za pomocą YaST 56
Wybór strefy czasowej 57
Ustawienia pulpitu 57
Wstępne ustawienia instalacji 57
Podział dysków na partycje 58
Instalacja pakietów oprogramowania 59
Tworzenie użytkownika root 59
Konfiguracja sieci i połączeń internetowych 60
Tworzenie użytkownika 62
Konfiguracja sprzętu dla X Window System 63
Logowanie i wyłączenie systemu po raz pierwszy 64
Odnośniki 66

Rozdział 4. Dalsza konfiguracja za pomocą YaST i SaX2 67
Rozwiązywanie problemów z instalacją 67
Konfiguracja sprzętu za pomocą YaST 68
Gromadzenie informacji o sprzęcie 68
Zmiana ustawień kontrolera dysków 68
Konfiguracja joysticka 69
Dostosowanie układu klawiatury 70
Konfiguracja myszy 70
Konfiguracja skanera 70
Konfiguracja karty TV lub radiowej 71
Ustawienia daty i godziny 71
Utrzymanie dokładnego czasu za pomocą NTP 73
Zarządzanie zasilaniem 77
Zarządzanie energią w YaST 77
Problemy z komputerami przenośnymi 77
Opcje zasilania w laptopach 78
Współpraca z kartami PCMCIA 79
Konfiguracja obsługi Bluetooth 79
Konfiguracja trybów pracy IrDA 80
Konfiguracja CD i DVD 81
Integrowanie napędów CD i DVD z systemem 81
Przyspieszanie pracy CD za pomocą DMA 82
Odnośniki 83

Część II Praca z systemem SUSE Linux 85

Rozdział 5. Rozpoczęcie pracy 87
Logowanie do systemu 87
Logowanie z powłoki 88
Wylogowanie z powłoki 88
Korzystanie z linuksowego systemu plików 88
Najważniejsze polecenia systemowe z katalogów /bin i /sbin 90
Korzystanie z katalogów /boot i /dev 90

Edycja plików konfiguracyjnych w katalogu /etc 91
Katalog /home: miejsce na wszelkie dobra 92
Znajdowanie programów pulpitu w katalogu /opt 93
Katalog /usr: miejsce na większość programów 93
Dane ulegające zmianom: katalog /var 93
Zmiany informacji o użytkownikach i program finger 94
Wybór i korzystanie z powłoki 95
Wygodne cechy wspólne dla wszystkich powłok 95
Praca z bash 96
Praca z tcsh 98
Praca z zsh 98
Zarządzanie plikami za pomocą narzędzia Midnight Commander 99
ksh i ash 100
Wybór powłoki 101
Edycja tekstu 103
Praca z edytorem vim i jego klonami 104
Praca z edytorem Emacs i jego klonami 105
Kilka innych edytorów 105
Dlaczego nie mogę zmienić tego pliku? Sposób działania uprawnień 106
Zmiany uprawnień 108
Uprawnienia do katalogów 109
Praca z konta root 110
Tworzenie i usuwanie użytkowników 111
Wyłączanie i ponowne uruchamianie systemu 112
Odnośniki 113
System plików 113
Powłoki 113
Edytory 114
Praca z konta root 114

Rozdział 6. Pulpit 115

Wprowadzenie do X Window System 116
Konfiguracja X w systemie SUSE Linux 116
Wybór środowiska pulpitu: KDE i GNOME 118
KDE: K Desktop Environment 120
GNOME: GNU Network Object Model Environment 122
Zarządzanie plikami z pulpitu 123
Narzędzie Konqueror w KDE 123
Narzędzie Krusader w KDE 126
Narzędzie Nautilus w GNOME 128
Konfiguracja środowiska pulpitu 129
Centrum sterowania KDE 129
Dostosowanie ustawień GNOME 131
Co to jest menedżer okien i do czego się przyda? 132
Po co używać menedżera okien? 133
Uruchamianie systemu z menedżerem okien 133
fvwm 133
WindowMaker 134
IceWM 136
Odnośniki 137

Rozdział 7. Drukowanie w SUSE Linux 139
Wprowadzenie 139
Common Unix Printing System (CUPS) 140
Konfiguracja drukarek w YaST 141
Konfiguracja CUPS i drukarek sieciowych za pomocą KDEPrint 145
Drukowanie za pomocą CUPS 148
Praca z dokumentami PDF 149
Sterowanie drukowaniem z konsoli 151
Podstawowe polecenia drukowania 151
Unikanie problemów z drukowaniem 152
Urządzenia wielofunkcyjne 152
USB i starsze drukarki atramentowe 153
Odnośniki 153

Rozdział 8. Praca z powłoką 155
Potęga wiersza poleceń 156
Dopasowywanie wzorców w powłoce 157
Przekierowywanie wejścia i wyjścia 159
Potokowanie danych 159
Przetwarzanie w tle 160
Skrypty powłoki: wprowadzenie 160
Pisanie skryptów w powłoce bash 161
Zmienne w skryptach powłoki 163
Przypisywanie wartości do zmiennych 163
Dostęp do wartości zmiennych 163
Parametry pozycyjne 164
Pozyskiwanie zmiennych wiersza polecenia za pomocą parametrów pozycyjnych 165
Automatyzacja zadań za pomocą skryptów 165
Zmienne wbudowane 166
Odnośniki 167

Rozdział 9. Produktywność: pakiety biurowe i inne narzędzia 169
Porównanie pakietów biurowych dla systemu Linux 169
Praca z OpenOffice.org 170
Praca z KOffice 176
Praca z GNOME Office 179
Praca z Softmaker Office 180
Inne pakiety komercyjne 182
Komunikacja z PDA 183
Oprogramowanie wiersza poleceń PDA dla SUSE Linux 183
Zarządzanie własnymi finansami 184
Inne narzędzia zwiększające produktywność 185
Komputerowe przygotowywanie publikacji w programie Scribus 186
Tworzenie wykresów i diagramów blokowych w programie Dia 188
Mapy myśli w programie VYM 188
Aplikacje dla faksów i skanerów 189
Odnośniki 191

Rozdział 10. Obraz, dźwięk i inne rozrywki 193
Konfiguracja kart dźwiękowych i wideo 193
Konfiguracja karty dźwiękowej i graficznej w YaST 194
Odtwarzanie muzyki i słuchanie radia 194
Formaty dźwięku 195
Radio internetowe i inne media przesyłane strumieniowo 196
Tworzenie własnej muzyki 197
Nagrywanie CD i DVD 197
Tworzenie CD i DVD z pulpitu 199
Tworzenie CD z powłoki 201
Tworzenie DVD z powłoki 202
TV i wideo — MythTV 204
Sprzęt TV i wideo 204
Linux i aparaty cyfrowe 207
Kamery internetowe 208
Edycja obrazów w programie GIMP 208
Formaty plików graficznych 211
Gry w systemie Linux 211
Gry w KDE 212
Gry w GNOME 213
Freeciv: darmowa gra strategiczna 213
Odnośniki 214

Rozdział 11. Rozwiązania wieloplatformowe 217
Przeglądanie plików po stronie systemu Windows 217
Emulacja innych systemów operacyjnych w Linuksie — wprowadzenie 219
Kilka nieporozumień na temat narzędzi wieloplatformowych 219
Uruchamianie aplikacji Windows za pomocą Wine 219
Instalacja i konfiguracja Wine 220
Instalacja aplikacji Windows w Wine 221
Praca z Wine 222
Crossover Office 222
Uruchamianie aplikacji Windows za pomocą Win4Lin 223
Instalacja Win4Lin 224
Praca z Win4Lin 224
Uruchamianie aplikacji Windows za pomocą VMware 225
Instalacja VMware 226
Praca z VMware 226
Emulacja systemu Mac OS za pomocą Basilisk II 228
dosemu i DOSBox 229
Xen: przyszłość wirtualizacji? 229
Odnośniki 232

Część III Korzystanie z Internetu 233

Rozdział 12. Łączenie z Internetem 235
Podstawowe informacje o nawiązywaniu łączności 235
Na początek interfejs localhost 236
Ręczna konfiguracja localhost 237

Konfiguracja modemowego połączenia z Internetem 238
Konfiguracja połączenia modemowego w YaST 239
Konfiguracja połączenia DSL (ang. Digital Subscriber Line) z Internetem 243
Konfiguracja połączenia DSL w YaST 244
Łączenie komputera z siecią bezprzewodową 247
Przygotowanie oprogramowania sprzętowego i sterowników 248
Szukanie punktu dostępowego 248
Konfiguracja karty bezprzewodowej 249
Kiedy skontaktować się z ISP? 250
Odnośniki 251

Rozdział 13. Przeglądanie WWW i obsługa poczty elektronicznej 253

Wybór przeglądarki WWW 253
Mozilla i Mozilla Firefox 254
Konqueror 256
Epiphany 257
Galeon 258
Opera 258
Konfiguracja konta pocztowego w SUSE Linux 259
Wybór klienta poczty 261
Evolution 262
KDE Personal Information Manager: Kmail/KOrganizer i Kontact 266
Mozilla Mail i Thunderbird 269
Sylpheed 270
mutt 270
Wysyłanie plików w załącznikach 272
BinHex 273
yEnc 273
uuencode/uudecode 273
MIME/Base64 274
Dekodowanie załącznika za pomocą UUDeview 274
Odnośniki 274

Rozdział 14. Tworzenie prostych serwisów WWW 277

Wybór narzędzia do tworzenia serwisu WWW 277
Dla początkujących 277
Narzędzia do tworzenia stron oparte na przeglądarkach 279
Wyspecjalizowane narzędzia WWW 280
Narzędzia WWW dla ekspertów 282
Ćwiczenie praktyczne: tworzenie serwisu w Quanta Plus 282
Prowadzenie weblogów w systemie SUSE Linux 289
Blosxom 290
Movable Type i Typepad 291
Blogi w WordPress 292
Popularne serwisy blogów 293
Dystrybucja blogów 293
Odnośniki 295

Rozdział 15. Zarządzanie serwerami poczty 297
Jak działa poczta elektroniczna? 297
Konfiguracja MTA 299
Maskarada 299
Autoryzacja 299
Zdalne połączenia SMTP 300
Protokoły poczty przychodzącej 300
Tryb doręczania 300
Aliasy 301
Domeny wirtualne 301
Walka ze spamem i wirusami 301
Narzędzia antyspamowe 302
Filtrowanie spamu po stronie klienta 303
Procmail 303
Narzędzia antywirusowe 304
Alternatywy dla programu Microsoft Exchange Server 304
Microsoft Exchange Server 305
CommuniGate Pro 305
Novell GroupWise 305
Samsung Connect 306
OPEN-XCHANGE 306
Kolab/Kroupware 306
OpenGroupware (Ogo) 306
phpGroupWare 306
Hula 307
Podsumowanie 307
Odnośniki 307

Rozdział 16. Współpraca z innymi użytkownikami 309
Usenet dzisiaj 310
Wybór czytnika grup dyskusyjnych 312
Klienty poczty i wiadomości 312
Samodzielne graficzne czytniki grup dyskusyjnych 313
Tekstowe czytniki grup dyskusyjnych 316
Współpraca z Wiki 317
ZWiki 317
Prowadzenie elektronicznych list dystrybucyjnych 320
Instalacja programu Mailman 321
Prowadzenie listy w programie Mailman 321
Korzystanie z komunikatorów internetowych i IRC 322
Internet Relay Chat (IRC) 323
Komunikatory w Linuksie 324
Wideokonferencje w systemie Linux 325
Odnośniki 326

Rozdział 17. Bezpieczne przesyłanie plików 329
Wybór klienta FTP 330
Tekstowe klienty FTP 330
Bezpieczne przesyłanie plików za pomocą sftp 332
Graficzne klienty FTP 333

Przesyłanie plików w P2P 336
Udostępnianie plików w stylu usługi Napster 336
BitTorrent 340
Konfiguracja serwera FTP 342
Konfiguracja xinetd do bezpiecznego przesyłania plików 342
Konfiguracja serwerów do bezpiecznego przesyłania plików 343
Odnośniki 344

Część IV Podstawy zarządzania systemem 347

Rozdział 18. Zarządzanie plikami, woluminami i dyskami 349
Wybór systemu plików 350
Fizyczna struktura dysku 351
Praca z systemem plików ReiserFS 351
Extended File System (ext2 i ext3) 352
JFS i XFS 353
Wybór systemu plików 353
Tworzenie systemów plików za pomocą narzędzia Expert Partitioner i zarządzanie nimi 353
Praca z narzędziem Expert Partitioner 354
Montowanie systemu plików 360
Polecenie mount 360
Polecenie umount 361
Automatyczne montowanie systemów plików za pomocą /etc/fstab 361
Logical Volume Management (LVM) 362
Znajdowanie plików 363
Polecenie find 363
Polecenie locate 364
Polecenie which 365
Polecenie whereis 366
Narzędzie KFind 366
Beagle 368
Manipulowanie systemami plików 371
Tworzenie systemu plików do testów 371
Montowanie partycji tylko do odczytu w uruchomionym systemie 373
Dostęp do pliku obrazu dyskietki 374
Zarządzanie plikami urządzeń znakowych, blokowych i specjalnych 375
Odnośniki 377

Rozdział 19. Zarządzanie użytkownikami i zabezpieczeniami 379
Konta użytkowników i uprawnienia 379
Tworzenie kont użytkowników w YaST 380
Konta użytkowników systemowych 383
RWX: podstawy uprawnień 383
Kto czego potrzebuje? Zarządzanie grupami 384
Dodawanie grup 384
Hasła: pierwsza linia obrony 386
Pliki passwd i shadow 386
Wybór haseł 387
Ustawienia uwierzytelniania użytkowników w PAM 388

Monitorowanie działań użytkowników 390
Narzędzia powłoki służące do obserwacji użytkowników 390
Superużytkownicy: pozwolić śmiertelnikom bawić się magią 391
Zmiana tożsamości użytkownika poleceniem su 392
Przyznawanie przywilejów konta root za pomocą polecenia sudo 392
Odnośniki 394

Rozdział 20. Zarządzanie danymi: tworzenie i przywracanie kopii zapasowych 395

Przygotowanie do ochrony przed utratą danych 395
Jak można stracić dane? 396
Szacunek potrzeb i dostępnych zasobów 396
Wybór strategii obsługi kopii zapasowych 397
Wybór nośnika 399
Dyski lustrzane i macierze RAID 399
Nośniki wymienne 400
Sieciowe magazyny danych 401
Taśmowe kopie zapasowe 402
Archiwizacja systemu w YaST 402
Oprogramowanie do obsługi kopii zapasowych 406
Archiwizacja z użyciem narzędzia tar 407
Korzystanie z narzędzia Ark 408
GNOME File Roller 409
Konfiguracja dysków lustrzanych i macierzy RAID 410
Ratowanie uszkodzonego systemu 413
Uruchamianie z płyty DVD SUSE Linux 413
Uruchamianie z zestawu dyskietek ratunkowych 414
Odtwarzanie systemu 414
Odnośniki 416

Rozdział 21. Aktualizacje systemu: zarządzanie pakietami 419

Instalacja oprogramowania za pomocą YaST 419
Podstawy systemu RPM 425
Pobieranie pakietów RPM spoza dystrybucji SUSE 426
Aktualizacja systemu za pomocą YaST Online Update (YOU) 427
Korzystanie z YOU 427
Monitorowanie serwerów YOU za pomocą narzędzia SUSEwatcher 430
Aktualizacje systemu za pomocą APT 432
Aktualizacje systemu za pomocą Red Carpet 434
Instalacja Red Carpet 435
Używanie Red Carpet z wiersza poleceń (rug) 435
Aktualizacje za pomocą interfejsu graficznego Red Carpet 436
Kompilacja oprogramowania z kodu źródłowego 437
Odnośniki 439

Część V Zaawansowane zarządzanie systemem 441

Rozdział 22. Zarządzanie startem systemu i innymi usługami 443

Proces uruchamiania systemu — wprowadzenie 443
Początek procesu — z BIOS-u do menedżera startowego 444
Wybór menedżera startowego: GRUB czy LILO? 444
Ładowanie jądra 448
Usługi i poziomy pracy systemu 449
Wybór usług przy starcie systemu za pomocą narzędzia YaST Usługi systemowe 453
Praca z narzędziem Usługi systemowe 454
Tryb zaawansowany 454
Ręczne uruchamianie i zatrzymywanie usług 455
Narzędzia do monitorowania systemu 456
KDE Strażnik Systemu 456
KDiskFree i KwikDisk 458
GKrellM 458
Odnośniki 459

Rozdział 23. Zabezpieczanie komputerów 461

Ataki na systemy komputerowe — wprowadzenie 462
Oszacowanie podatności na atak 463
Ochrona komputera 465
Zabezpieczanie sieci bezprzewodowej 466
Jeszcze kilka słów na temat haseł i fizycznych zabezpieczeń 466
Konfiguracja SUSE Firewall 467
Inne narzędzia godne uwagi 469
Co zrobić po włamaniu? 469
Nadążanie za problemami z bezpieczeństwem Linuksa 471
Odnośniki 471

Rozdział 24. Zarządzanie jądrem i modułami 473

Linus i jego dzieło: jądro Linuksa 474
Drzewo kodu źródłowego jądra 474
Typy jądra 474
Zarządzanie modułami 475
Kiedy rekompilować? 476
Wersje jądra 477
Pobieranie nowego kodu źródłowego jądra 478
Instalowanie poprawek do jądra 480
Kompilacja jądra 481
Rozwiązywanie problemów 482
Błędy podczas kompilacji 482
Błędy wykonania, problemy z programem rozruchowym i błędy kernel oops 483
Dostrajanie jądra za pomocą sysctl 484
Odnośniki 485

Rozdział 25. Konfiguracja sieci i usługi Samba 487

Sieci TCP/IP 487
Adresy TCP/IP 488
Następny krok: adresowanie w IPv6 489
Porty 490

Organizacja sieci 490
Podsieci 491
Maski podsieci 491
Translacja adresów sieciowych (NAT) 491
Sprzęt sieciowy 491
Karty sieciowe 491
Kable sieciowe 493
Koncentratory 493
Inicjalizacja nowych urządzeń sieciowych 493
Protokół DHCP 496
Jak działa DHCP? 496
Instalacja i konfiguracja oprogramowania DHCP 496
Konfiguracja serwera DHCP 496
Network File System 497
Uruchomienie i konfiguracja serwera NFS 498
Importowanie katalogów w kliencie NFS 499
Współpraca z Windows za pomocą usługi Samba 499
Zabezpieczanie usług sieciowych 500
Odnośniki 500

Rozdział 26. Zarządzanie własnym serwerem WWW Apache 503

Co to jest Apache? 503
Instalacja Apache 504
Kompilacja Apache z kodu źródłowego 504
Uruchamianie i zatrzymywanie Apache 506
Ustawienia konfiguracji działania serwera 507
Dodatkowe ustawienia w pliku httpd.conf 509
Uwierzytelnianie i kontrola dostępu do systemu plików 509
Ograniczanie dostępu przez allow i deny 509
Uwierzytelnianie 510
Moduły Apache 512
Hosting wirtualny 513
Hosting wirtualny oparty na nazwach 514
Hosting wirtualny oparty na adresach IP 515
Treść dynamiczna 516
Programy CGI 516
SSI 517
Moduły materiałów aktywnych 517
Inne serwery WWW 518
thttpd 518
Caudium WebServer 518
Xitami 518
Zope 519
ColdFusion 519
TomCat 519
Odnośniki 519

Rozdział 27. Zarządzanie nazwami domen 521
Podstawowe pojęcia DNS 521
Hierarchia DNS 523
Strefy 526
Rekordy 526
Konfiguracja DNS w YaST2 527
Plik hosts 529
Podstawowe narzędzia DNS 529
Konfiguracja serwerów nazw w BIND 531
Rejestrowanie zdarzeń 534
Podsumowanie 535
Odnośniki 536

Część VI Programowanie 537

Rozdział 28. Korzystanie z GCC i innych narzędzi programistycznych 539
Podstawy programowania w C pod Linuksem 540
Prosty program w języku C 540
Korzystanie z GNU Compiler Collection 541
Budowanie aplikacji Java za pomocą Eclipse 543
Korzystanie z Eclipse Workbench 543
Tworzenie małej aplikacji Java w Eclipse 545
Zintegrowane środowisko programistyczne KDevelop 546
Tworzenie aplikacji dla GNOME za pomocą narzędzia Anjuta 549
Zarządzanie wieloosobowymi projektami programistycznymi za pomocą CVS i Subversion 551
Tworzenie pakietów RPM 553
Tworzenie pliku specyfikacji 553
Tworzenie listy plików i generowanie poprawek 555
Budowanie i testowanie pakietu RPM 556
Inne zasoby programistyczne dla Linuksa 556
Odnośniki 557

Rozdział 29. Zarządzanie bazami danych 559
Co to jest relacyjna baza danych? 559
Implementacja MySQL 561
Instalacja MySQL 561
Wstępna konfiguracja MySQL 563
Korzystanie z MySQL Control Center 566
Łączenie się z bazą danych MySQL za pomocą klienta baz danych 568
Korzystanie z klienta MySQL Monitor 568
Optymalizacja MySQL 572
Pomiary wykorzystania bufora kluczy 572
Korzystanie z pamięci podręcznej zapytań 574
Różnorodne poprawki 574
Optymalizacja zapytań 575
Podsumowanie 575
Odnośniki 576

Rozdział 30. Języki Perl i Python 577
Czym jest Perl? 578
Zmienne i struktury danych Perla 578
Operatory i inne instrukcje 579
Wyrażenia regularne 580
Moduły Perla i CPAN 582
Przyszłość języka Perl 583
Czym jest Python? 584
Zmienne i struktury danych języka Python 584
Wcięcia 585
Ekstremalna obiektowość 585
IDE i interaktywna powłoka Pythona 586
Odnośniki 586

Rozdział 31. Tworzenie dynamicznych serwisów WWW 589
Korzystanie z PHP 589
Instalacja PHP 590
Pisanie skryptów w PHP 591
Dodawanie funkcji do PHP 594
Przyszłość z PHP5 594
Tworzenie interaktywnych serwisów w WordPress 595
Instalacja WordPress 595
Nadzór nad dostępem użytkowników do WordPress 596
Tworzenie forum użytkowników opartego na WWW 597
Instalacja phpBB 597
Konfiguracja phpBB i tworzenie forów 598
Odnośniki 601

Rozdział 32. Optymalizacja wydajności 603
Optymalizacja usług 604
Powertweak 606
Dostrajanie wydajności napędów dyskowych 608
Wybór najlepszego sprzętu 609
Dostrajanie parametrów dysków przy starcie systemu 610
Polecenie hdparm 611
Wyłączanie rejestracji czasu dostępu do pliku 613
Rozkładanie obciążenia 613
Odnośniki 613

Rozdział 33. Wiersz polecenia — kurs zaawansowany 615
Kilka podstawowych poleceń 616
Wyświetlanie zawartości pliku poleceniem cat 617
Tworzenie i zmiana katalogów poleceniami mkdir i cd 618
Zmiana uprawnień dostępu do plików poleceniem chmod 619
Kopiowanie, przenoszenie i zmiana nazw plików za pomocą cp i mv 619
Wyświetlanie wykorzystania dysku poleceniem du 620
Szukanie łańcuchów poleceniem grep 620
Znajdowanie plików z indeksu poleceniem locate 621
Wyświetlanie plików w bieżącym katalogu poleceniem ls 621
Czytanie dokumentacji systemowej poleceniem man 622

Wyświetlanie listy procesów poleceniem ps 622
Usuwanie plików i katalogów poleceniem rm 623
Wyświetlanie ostatnich wierszy pliku poleceniem tail 623
Wyświetlanie stopnia wykorzystania zasobów poleceniem top 624
Wyświetlanie położenia programu poleceniem which 624
Łączenie poleceń 624
Odnośniki 625

Dodatki 627

Dodatek A Wersje SUSE Linux firmy Novell 629

Dodatek B SUSE i Linux — zasoby internetowe 633

Skorowidz 661

Rozdział 8.
Praca z powłoką

„Na początku był wiersz poleceń". Tako rzecze Neal Stephenson w swoim rewelacyjnym eseju na temat historii komputerowych systemów operacyjnych. I tak też jest w rzeczywistości — w Linuksie wiele rzeczy najłatwiej i najszybciej można zrobić wprost z wiersza poleceń, zwanego **powłoką** (ang. *shell*). Dlaczego „powłoka"? W języku angielskim orzech składa się z jądra (ang. *kernel*) i skorupy (ang. *shell*). Wprawdzie w języku polskim „shell" oznacza „powłokę", jednakże w tym kontekście jest też „skorupą otaczającą jądro systemu".

Interfejs graficzny typu „wskaż i kliknij" jest łatwy w użyciu. Z drugiej strony, potęga wiersza poleceń i zdolność do wykonywania wielu zadań przez naciśnięcie kilku klawiszy jest tak duża, że czasem trudno sobie wyobrazić narzędzie łatwiejsze w użyciu. Problem tkwi w krzywej uczenia się. W interfejsie graficznym (GUI) wystarczy przyzwyczaić się do kilku zasad używania myszy i można wykonać większość zadań. W przeciwieństwie do GUI puste okno wiersza zachęty powłoki nie zawiera żadnych wskazówek, jak wykonywać określone prace.

Nawet największy ekspert od graficznego interfejsu użytkownika może potrzebować przywrócenia systemu, a wtedy do dyspozycji otrzymuje tylko wiersz poleceń. Znajomość kilku poleceń jest konieczna, by jakoś sobie z nim poradzić. W całej niniejszej książce Czytelnik spotka mnóstwo pomysłów, jak za pomocą wiersza poleceń szybko wykonać różne działania.

W niniejszym rozdziale skoncentrujemy się na powłoce jako języku programowania. Większość powłok zawartych w dystrybucji SUSE Linux pozwala łączyć polecenia w jeden ciąg w edytorze tekstu i zapisywać wynik jako skrypt. Znając składnię powłoki, możemy definiować zmienne, przypisywać wartości, tworzyć funkcje i ewaluować różne wyniki — robić rzeczy dostępne w „prawdziwych" językach programowania, takich jak C, Java lub C++. Wiele standardowych poleceń powłoki jest w rzeczywistości skryptami. Być może Twój skrypt też któregoś dnia będzie w powszechnym użyciu.

Potęga wiersza poleceń

Wiedza o możliwościach powłoki jest pierwszym krokiem do pisania dobrych skryptów. Powłoka jest zdecydowanie czymś więcej niż narzędziem do uruchamiania programów, aczkolwiek tę funkcję pełni bez zarzutu. Powłokę nazywa się też „interpreterem poleceń", ponieważ przyjmuje polecenie wpisane przez użytkownika i robi z nim kilka rzeczy:

1. W pierwszej kolejności ustala, czy wpisane polecenie jest w nią wbudowane. Jeśli tak, wykonuje je.
2. Jeśli polecenie nie jest wbudowane, posługuje się zmienną środowiskową ścieżki dostępu, aby znaleźć polecenie (o ile nie wpisaliśmy go z podaniem pełnej ścieżki).
3. Po zlokalizowaniu polecenia powłoka wywołuje funkcję `fork` w jądrze Linuksa, aby skopiować powłokę (dzięki czemu po ukończeniu pracy przez program powróci do wiersza zachęty). Program następnie wywołuje kolejną funkcję jądra `exec`, aby wykonać polecenie.
4. Na koniec powłoka na podstawie określonych reguł ustala, co polecenie, np. `cat *.txt *.sxw > doclist` ma osiągnąć (tu odbywa się interpretacja) i dostarcza wynik.

Powłoka może uruchomić kilka poleceń w jednym ciągu, jak w powyższym przykładzie. Przykład ten jest stosunkowo prosty, lecz powłoka wykonuje w nim kilka działań:

- Uruchamia program `cat`, który wyświetla pliki na ekranie.
- Przeprowadza sterowanie zadaniem i usuwa się z drogi — powłoka „udaje", że jej nie ma, gdy program `cat` szuka tekstu wszystkich plików z rozszerzeniami *.txt* i *.sxw* w bieżącym katalogu, a następnie go wyświetla.
- Przekierowuje wyjście — zanim tekst trafi na ekran, powłoka z powrotem wkracza do akcji i przechwytuje funkcję systemową, umieszczając wyświetlane informacje (tzn. tekst wszystkich plików ze wspomnianymi rozszerzeniami) w pliku o nazwie *doclist* i zapisując ten plik w tym samym katalogu.

Uruchamianie poleceń powłoki w narzędziu Konqueror

Konqueror — menedżer plików KDE — pozwala połączyć potęgę wiersza poleceń z wygodą interfejsu graficznego i wydawać dowolne polecenia w swoim interfejsie.

Należy otworzyć *Narzędzia*/*Wykonaj polecenie powłoki* (albo nacisnąć *Ctrl+E*). Jeśli zaznaczona została nazwa pliku lub katalogu, to pojawi się w oknie polecenia; nie trzeba jednak wykonywać polecenia na tym akurat pliku. Każde wpisane polecenie (lub ciąg poleceń) zostanie uruchomione; komunikaty będą wyświetlane w osobnym oknie.

Jeśli Konqueror został otwarty z konta superużytkownika, wiersz poleceń udostępni wszystkie przywileje tego konta, lecz zwykłe okno Konqueror nie pozwala zalogować się jako superużytkownik (ani uruchamiać poleceń wymagających wprowadzenia dodatkowych informacji).

Do możliwości powłoki (oprócz listy „Wygodne cechy wspólne dla wszystkich powłok" z rozdziału 5., „Rozpoczęcie pracy") należą:

- Szukanie w plikach i katalogach wzorców (wyrażeń regularnych) za pomocą narzędzi `locate`, `gawk` i `grep`.
- Kierowanie w celu dalszego przetwarzania wyjścia jednego programu na wejście innego programu.
- Uruchamianie w jednym wierszu kilku poleceń oddzielonych średnikami.
- Tworzenie kilku pustych plików jednym poleceniem:

```
touch file {a,b,c,d,e}
```

- Uruchamianie programów w tle.

Wszystkie te funkcje przydadzą się przy pisaniu skryptów, omówionych w niniejszym rozdziale.



Przykłady przedstawione tutaj są przeznaczone dla powłoki bash, która jest zdecydowanie najpopularniejszą (i domyślną) powłoką w systemie SUSE Linux. Skrypty wykonujące praktycznie wszystkie zadania opisane poniżej można pisać również w innych powłokach — składnia powinna być opisana na stronach man powłok. Czytelnik może też skorzystać z książek *Sams Teach Yourself Shell Programming in 24 Hours* (autor: Sriranga Veeraraghavan) lub *Linux Shell Scripting with Bash* (Ken O. Burtch).

Dopasowywanie wzorców w powłoce

Wiersz poleceń powłoki pozwala używać łańcuchów specjalnie skonstruowanych wzorców, aby dopasowywać wieloznaczne ciągi. Jest to funkcjonalność inna i prostsza niż obsługiwana przez narzędzia GNU typu grep (używane są w nich bardziej złożone wzorce, zwane wyrażeniami regularnymi) i służy do przeszukiwania plików i katalogów albo filtrowania danych kierowanych do i z poleceń.

Łańcuchy wzorców w powłoce mogą być proste lub skomplikowane, lecz w wierszu poleceń, nawet posługując się skromnym podzbiorem dostępnych symboli wieloznacznych, możemy uzyskać konstruktywne wyniki. Do symboli wieloznacznych najczęściej używanych w powłoce należą:

***** — dopasowuje dowolny ciąg znaków. Na przykład, aby znaleźć w bieżącym katalogu wszystkie pliki o rozszerzeniu *.txt*, możemy użyć polecenia:

```
ls *.txt
```

? — dopasowuje pojedynczy znak. Na przykład, aby znaleźć w bieżącym katalogu wszystkie pliki o rozszerzeniu *d?c*, gdzie zamiast ? może wystąpić litera a – z, A – Z lub cyfra 0 – 9, możemy użyć polecenia:

```
ls *.d?c
```

[*XXX*] lub **[*X*-*X*]** — dopasowuje przedział znaków. Na przykład, aby wyświetlić wszystkie pliki z bieżącego katalogu mające nazwy zaczynające się od cyfry, możemy użyć polecenia:

```
ls [0-9]*
```

Aby znaleźć tylko pliki o nazwach zaczynających się od 1, 5 lub 8, użyjemy polecenia:

```
ls [158]*
```

`\X` — dopasowuje lub rozpoczyna sekwencję unikową dla znaku specjalnego, na przykład `?`. Poniższe polecenie tworzy plik o nazwie zawierającej znak zapytania:

```
touch foo\?
```

Tabela 8.1 przedstawia kilka innych popularnych znaków specjalnych. Bardziej wyczerpującą listę zawiera dokument *Advanced Bash Scripting Guide*, wymieniony w podrozdziale „Odnośniki".

Tabela 8.1. *Znaki specjalne w powłoce bash*

Znak	Znaczenie
#	Początek wiersza komentarza (z wyjątkiem wiersza `#!`).
$	Nazwa zmiennej powłoki.
;	Separator poleceń.
.	Katalog bieżący.
..	Katalog nadrzędny.
~	(tylda) Katalog macierzysty użytkownika. Znak mieści się na klawiszu po lewej stronie klawisza cyfry 1; uzyskasz do niego dostęp, przytrzymując klawisz *Shift*.
`	Podstawienie polecenia; mówi powłoce, że wyniki najbliższego polecenia mają zostać przechwycone w celu ich dalszego użycia.
>	Przekierowanie wyjścia skryptu do pliku. Jeśli plik istnieje, zostanie zastąpiony.
>>	Dołącza wyjście skryptu do istniejącego pliku. Jeśli plik nie istnieje, zostanie utworzony.
|	Potok. Pomaga łączyć w łańcuch polecenia i skrypty, przekierowując wyjście jednego polecenia na wejście następnego.
&	Wskazane polecenie ma zostać uruchomione w tle. Pozwala powrócić do wiersza zachęty powłoki.

Używając znaków specjalnych, należy zawsze uważać. Jak widać, niektóre znaki specjalne mają różne znaczenia zależnie od kontekstu, skrypt może zatem zachowywać się niewłaściwie, jeśli znak nie zostanie użyty poprawnie. Niektóre znaki mogą być nawet szkodliwe, wobec czego musimy upewnić się, że używamy właściwych.

Aby użyć znaku specjalnego jako „zwykłego", należy poprzedzić go w skrypcie znakiem unikowym, którym w tym przypadku jest lewy ukośnik (`\`). Na przykład, poniższe polecenie wyświetla wynik obliczenia kwoty w dolarach:

```
echo $((5153 + 654863))
```

Mieszanie w skrypcie symboli wieloznacznych z wyrażeniami regularnymi może przy odrobinie nieuwagi powodować problemy. Na przykład, do wyszukiwania wzorców w tekście najlepiej nadają się wyrażenia regularne, używane w poleceniach typu `grep`; proste symbole wieloznaczne powinny być używane np. do filtrowania i dopasowywania nazw plików w wierszu poleceń. Wprawdzie zarówno w wyrażeniach regularnych, jak i skryptach powłoki w Linuksie lewy ukośnik rozpoznawany jest we wzorcach jako znak unikowy, jednak symbol dolara (`$`) ma dwa całkowicie odmienne znaczenia (w wyrażeniach regularnych dopasowywanie pojedynczego znaku, a w skryptach przypisanie zmiennej).

Przekierowywanie wejścia i wyjścia

Wiersz poleceń pozwala tworzyć, nadpisywać i dodawać dane do plików za pomocą procesu zwanego przekierowaniem wyjścia. Powłoka rozpoznaje kilka znaków specjalnych stosowanych w tym procesie, na przykład >, < i >>.

W poniższym przykładzie wyjście polecenia `ls` zostaje przekierowane, aby utworzyć plik o nazwie *textfiles.listing*:

```
ls *.txt > textfiles.listing
```

Przekierowania wyjścia należy używać ostrożnie, ponieważ z łatwością można nadpisać istniejące pliki: wystarczy przekierować polecenie do istniejącego pliku. Powłoka wykona żądane zadanie bez sprawdzenia, czy plik już istnieje i bez pytania, co zrobić z plikiem, z którym wkrótce się pożegnamy.

Na szczęście można też przekierować wyjście tak, że dane zostaną dołączone do istniejącego pliku, używając operatora >>. Aby więc dodać coś ciekawego do naszej istniejącej kolekcji plików tekstowych, użyjemy dwóch znaków > zamiast jednego:

```
ls /usr/share/doc/packages/emacs/doc/gnus-tut.txt >> textfiles.listing
```

Nawias trójkątny można też odwrócić, by pobrać dane i wprowadzić je do polecenia przez przekierowanie wejścia, jak w tym przykładzie:

```
cat < textfiles.listing
```

Potokowanie danych

Wiele poleceń linuksowych można uruchomić razem w jednym wierszu polecenia, aby przekształcić dane z jednej postaci w inną. Łączenie poleceń w ten sposób nazywane jest **potokowaniem** lub **pracą potokową**. Potok (ang. *pipe*) pobiera wyjście jednego polecenia i wprowadza jako wejście następnego. Ten proces można powtarzać niemal w nieskończoność.

W powłoce do tworzenia potoku służy symbol pionowej poprzeczki (|). Używany jest w tej roli tak często, że zyskał sobie potoczną nazwę „pipe". Załóżmy, że administrator systemu chce sprawdzić, kto w danej chwili jest zalogowany. Połączenie kilku poleceń potokowo, jak poniżej, zwróci uporządkowaną, czytelną listę:

```
who | cut -c1-8 | sort -u | pr -l1 -8 -w78 -t
```

Zostały tu połączone cztery polecenia:

1. `who` zwraca listę zalogowanych użytkowników.
2. Lista jest przetwarzana przez polecenie `cut`, które wyświetla na ekranie tylko nazwy użytkowników (zamiast całego wiersza wyjścia).

3. Jeszcze przed wyświetleniem listy polecenie `sort` sortuje nazwy użytkowników w porządku alfabetycznym i usuwa z listy powtarzające się wpisy użytkowników, którzy są zalogowani więcej niż raz.
4. Na koniec polecenie `pr` pobiera listę wiersz po wierszu i wyświetla ją na ekranie w ośmiu kolumnach.

Jest to prosta technika, a dysponując wystarczającym doświadczeniem, Czytelnik będzie mógł robić naprawdę zadziwiające rzeczy za pomocą kilku poleceń i kilku potoków.

Przetwarzanie w tle

W niniejszej książce często będą pojawiać się polecenia zakończone znakiem `&`, zwłaszcza w przypadku uruchamiania z powłoki aplikacji graficznych. Czytelnik może zastanawiać się, do czego to służy. Symbol `&` wykorzystuje zdolność powłoki do przetwarzania informacji w tle i do jednoczesnego uruchamiania innych programów z tego samego wiersza poleceń.

Załóżmy, że chcemy uruchomić z powłoki monitor systemu GKrellM. Po wpisaniu `gkrellm` w wierszu poleceń program oczywiście się uruchomi. Lecz gdy spojrzymy następnie na okno powłoki, zobaczymy „martwy" kursor, który nie pozwoli wprowadzić żadnego innego polecenia, przynajmniej do chwili wyjścia z programu GKrellM.

Po zakończeniu pracy GKrellM (przez kliknięcie prawym przyciskiem myszy dowolnego miejsca w oknie programu i wybraniu *Quit* z menu) wiersz zachęty powróci do okna powłoki i będzie w nim można ponownie wpisywać polecenia. Naciśnięcie klawisza strzałki w górę, aby przywołać ostatnio wydane polecenie z historii poleceń, wyświetli ponownie `gkrellm` w wierszu zachęty. Tym razem możemy dodać na koniec wiersza znak `&`. Po naciśnięciu *Enter* program GKrellM uruchomi się jak poprzednio, lecz tym razem powłoka bash zwróci numer procesu aplikacji i powróci do wiersza zachęty, w którym będzie można wpisywać kolejne polecenia.

Skrypty powłoki: wprowadzenie

Historia poleceń w wierszu zachęty jest bardzo wygodnym narzędziem, ale co zrobić, jeśli musimy codziennie wydawać podobne polecenia? Jeśli zachodzi potrzeba wykonania co tydzień skomplikowanego zestawu poleceń, lecz niekoniecznie w ten sam dzień tygodnia? Tutaj przydają się skrypty.

Skrypty powłoki nie są, co prawda, szybsze niż program w C, lecz nie wymagają kompilacji, a co za tym idzie, mogą być o wiele mniejsze. Dzięki temu są również bardziej przenośne: zawsze zmieszczą się na dyskietce. Nie wymagają instalowania w systemie dodatkowych bibliotek, o ile skrypt nie wymaga dostępu do innego języka. Proces pisania skryptów jest też prostszy i szybszy niż w przypadku języków kompilowanych.

Czytelnikom dopiero zaczynającym pisać skrypty powłoki radzę nie przesadzać. To trochę jak nowy program ćwiczeń. Nie należy próbować zrobić zbyt dużo za jednym razem, lecz skoncentrować się na automatyzacji używanego już procesu (albo takiego, który wydawał się zbyt skomplikowany, by go wykonywać).

Najważniejsze przy pisaniu skryptu jest to, by nigdy nie używać procesora tekstu! Tworzymy plik tekstowy, i standardowy edytor tekstu w zupełności wystarczy. Pomijając już dodawane do tekstu formatowanie, procesory tekstu stosują automatyczne zawijanie wierszy, które może z łatwością namieszać w skrypcie. Radzę przyjrzeć się edytorom przedstawionym w rozdziale 5. i poeksperymentować z nimi, by wybrać najlepszy dla siebie.



SUSE Linux automatycznie tworzy w katalogu macierzystym użytkownika podkatalog */bin*. Jest to dobre miejsce do przechowywania skryptów, ponieważ znajduje się na pierwszym miejscu w ścieżce wyszukiwania.

Pisanie skryptów w powłoce bash

Na początek możemy zdefiniować kilka aliasów poleceń, które uproszczą pewne zadania wykonywane w wierszu poleceń. Aliasy w Linuksie są zmiennymi środowiskowymi o zachowaniu takim, jakie sugeruje ich nazwa — stanowią inne nazwy poleceń. Aliasy można definiować na bieżąco w wierszu poleceń, lecz tak zdefiniowane trwają tylko do końca sesji. Lepiej, jeśli są dostępne dla użytkownika cały czas.

Przed rozpoczęciem pisania skryptu przyjrzyjmy się domyślnym aliasom zdefiniowanym przez SUSE. Wpisanie `alias` w wierszu poleceń wyświetli poniższy listing.

```
alias +='pushd .'
alias -='popd .'
alias ..='cd ..'
alias ...='cd ../..'
alias beep='echo -en "\007"'
alias dir='ls -'
alias l='ls -alF'
alias la='ls -la'
alias ll='ls -l'
alias ls='/bin/ls $LS_OPTIONS'
alias ls-l='ls -l'
alias mc='. /usr/share/mc/bin/mc-wrapper.sh'
alias md='mkdir -p'
alias o='less'
alias rd='rmdir'
alias rehash='hash -r'
alias unmount='echo "Error: Try the command: umount" 1>&2; false'
alias which='type -p'
alias you='su - -c "/sbin/yast2 online_update"'
```

Aliasy te są zdefiniowane w plikach ustawień środowiskowych o zasięgu całego systemu: */etc/profile* i */etc/bash.bashrc* i domyślnie obowiązują dla wszystkich użytkowników. Każdy użytkownik może dostosować własne aliasy, modyfikując plik *~/.bashrc*. Możemy napisać skrypt zawierający dowolne aliasy i odwołać się do niego w pliku *.bashrc*.

Na przykład, otwórz edytor tekstu i wpisz:

```
#!/bin/sh
alias ls='ls -l'
alias ldir='ls -aF'
alias copy='cp'
alias ft='fortune -a'
```

Znaczenie powyższego skryptu jest następujące:

- Wiersz 1. deklaruje, że mamy do czynienia ze skryptem powłoki (bash).
- Wiersz 2. mówi, że listing katalogu wyświetlany poleceniem `ls` ma zawsze mieć szczegółową (długą) postać, zawierającą informacje o uprawnieniach, rozmiarze, dacie i czasie. Wynikiem będzie taki listing, jak dla domyślnego aliasu `ll`.
- Wiersz 3. mówi, że po wpisaniu `ldir` ma zostać wyświetlona kolorowa lista plików i katalogów bez podawania szczegółów.
- Wiersz 4. zmienia zachowanie powłoki po wpisaniu DOS-owego polecenia `copy`: zamiast komunikatu o błędzie wykonuje zadanie kopiowania.
- Wiersz 5. definiuje skrót (`ft`) do polecenia wyświetlającego cytat losowy (ang. *fortune cookie*). Opcja `-a` mówi, że program `fortune` ma przeszukiwać wszystkie swoje kolekcje cytatów, co może doprowadzić do wyświetlenia czegoś nieoczekiwanego. Tej opcji należy używać ostrożnie.

Po dopisaniu do pliku wszelkich pożądanych aliasów zapisz go jako *~/bin/myaliases*. Rozszerzenie nazwy nie jest wymagane. Pozostały jeszcze dwa zadania. Po pierwsze, na razie jest to tylko plik tekstowy. Aby mógł stać się skryptem, musi być wykonywalny. Otwórz okno powłoki, przejdź do podkatalogu *~/bin* i wpisz `chmod +x myaliases`. Każdy użytkownik jest właścicielem wszystkich plików w swoim katalogu macierzystym, więc dzięki temu poleceniu plik skryptu staje się wykonywalny.



Od początku przygody z pisaniem skryptów warto przyzwyczaić się do pisania komentarzy i dokumentowania. Na początku każdego skryptu radzę umieścić opis jego przeznaczenia. Dla powyższego przykładu komentarz może wyglądać tak:

```
# Ten skrypt dodaje kilka aliasów wyświetlania
# katalogów do domyślnego zestawu.
```

Każdy wiersz komentarza powinien zaczynać się od znaku #; w przeciwnym razie powłoka będzie próbowała zinterpretować go jako polecenie.

Aby dodać zdefiniowane aliasy do standardowej konfiguracji, otwórz plik *~/.bashrc* w edytorze tekstu i dodaj na koniec pliku następujący wiersz:

```
~/bin/myaliases
```

Zapisz plik i wyjdź z edytora. Wyloguj się i ponownie zaloguj, a następnie otwórz okno powłoki i wpisz `alias`. Zdefiniowane aliasy powinny teraz znaleźć się na liście.

Zmienne w skryptach powłoki

Powłoka, będąc językiem programowania, może przetwarzać zmienne w skryptach. Dostępne są trzy zmienne, które można wykorzystywać w skryptach powłoki:

- Zmienne środowiskowe są składnikiem systemu. W powyższym przykładzie zastosowana została zmienna `PATH`. Użytkownik może definiować nowe zmienne środowiskowe, a wartości niektórych zmiennych (np. `PATH`) modyfikować za pomocą skryptów.
- Zmienne wbudowane, na przykład opcje poleceń, udostępnia system operacyjny; nie można ich modyfikować.
- Zmienne użytkownika są definiowane i używane przez autora skryptu.



Czytelnicy mający doświadczenie z innymi językami programowania zauważą pewną różnicę: w skryptach powłoki zmienne są beztypowe.

I odwrotnie, zaczynając naukę programowania od skryptów i przechodząc do innych języków programowania, Czytelnik będzie musiał nauczyć się identyfikować zmienne jako liczby, łańcuchy i inne typy danych istniejące w danym języku.

Przypisywanie wartości do zmiennych

Deklarowanie i inicjalizowanie zmiennych w powłoce bash jest proste. Aby użyć np. zmiennej o nazwie `lcount` do zliczania iteracji pętli, należy wpisać `lcount=0`.

Aby zapisać łańcuch w zmiennej, wystarczy jednocześnie ją zadeklarować i zidentyfikować:

```
myname=mikemc
```

Jeśli łańcuch zawiera spacje, należy użyć cudzysłowów:

```
myname="Michael McCallister"
```

Dostęp do wartości zmiennych

Dostęp do wartości zmiennej wymaga poprzedzenia jej nazwy znakiem dolara (`$`). Na przykład, aby wyświetlić ścieżkę dostępu, należy wpisać `$PATH`. Jeśli zmienna ma nazwę `var`, użyjemy zapisu `$var`.

Aby przypisać wartość zmiennej `var` do zmiennej `lcount`, należy wpisać:

```
lcount=$var
```

Parametry pozycyjne

Do skryptu powłoki można przekazać opcje z wiersza poleceń lub innego skryptu. Opcje te noszą nazwę parametrów pozycyjnych i mają specjalne nazwy, zdefiniowane przez system. Pisanie zaawansowanych skryptów powłoki wymaga umiejętności korzystania z parametrów pozycyjnych i pobierania wartości z wiersza polecenia.

Nazwami parametrów pozycyjnych są w istocie liczby; pierwszy parametr ma nazwę 1, następny 2, trzeci 3 i tak dalej. Dostęp do nich odbywa się przez `$1`, `$2` itd. Nie można wywołać 2 przed 1, lecz można pominąć 3, aby uzyskać wynik, jeśli 1 i 2 są obecne.

Załóżmy, że mamy skrypt powłoki wymagający dwóch parametrów: imienia i nazwiska. Jeśli podamy tylko imię (1), to w porządku. Jeśli tylko nazwisko (2), otrzymamy błąd.

Poniższy skrypt o nazwie *myname1* pobiera jeden parametr (imię) i wyświetla na ekranie:

```
#!/bin/sh
# skrypt wyświetlający imię
if [ $# -eq 0 ]
then
    echo "Imię nie zostało podane"
else
    echo "Podane imię to "$1
fi
```



W skryptach powłoki każda instrukcja `if` musi być zakończona instrukcją `fi`.

Uruchomienie `myname1` następująco:

```
bash myname1
```

zwróci wynik:

```
Imię nie zostało podane
```

Poniższe polecenie:

```
bash myname1 Mike
```

zwróci taki wynik:

```
Podane imię to Mike
```

Czytelnik może zastanawiać się, co w tym skrypcie oznacza kod `$#`. Jest to zmienna wbudowana, udostępniana przez jądro dla powłoki i zwracająca liczbę parametrów pozycyjnych, przekazanych do programu powłoki. Więcej o tych wbudowanych zmiennych powiemy poniżej.

Pozyskiwanie zmiennych wiersza polecenia za pomocą parametrów pozycyjnych

Parametry pozycyjne mogą być bardzo przydatne przy łączeniu potokowym poleceń z użyciem złożonych argumentów, zwłaszcza jeśli polecenia nie są często używane. Na przykład, system z modemem głosowym może służyć jako automatyczna sekretarka. W takim komputerze możemy napisać skrypt, który pobiera i odtwarza wiadomości. Skrypt ten, o nazwie *pvm*, konwertuje zapisany plik dźwiękowy (w formacie telefonu głosowego *.rmd*) i kieruje dźwięk do urządzenia audio:

```
#!/bin/sh
# odtwarzanie wiadomości głosowych z /var/spool/voice/incoming
rmdtopvf /var/spool/voice/incoming/$1 | pvfspeed -$ 0000; \
pvftobasic > /dev/audio
```

Aby odtworzyć wiadomość, należy wpisać w wierszu zachęty powłoki:

```
pvm <nazwa_wiadomości>
```

Automatyzacja zadań za pomocą skryptów

Skrypty powłoki z parametrami pozycyjnymi używane są często do automatyzacji rutynowych i zwykłych zadań, na przykład generowania raportów z dzienników zdarzeń, sprawdzania systemów plików, rozliczania wykorzystania zasobów przez użytkowników i innych zadań związanych z systemem, siecią lub bezpieczeństwem.

Poniższy skrypt po dodaniu do tablicy harmonogramu */etc/crontab* może pomóc w monitorowaniu systemu, szukając w plikach dziennika słów kluczowych wskazujących ważne informacje.

```
#!/bin/sh
# nazwa: greplog
# zastosowanie: wysyła pocztą wyniki szukania słowa kluczowego
# w dzienniku zdarzeń
#
# autor: bb
# sposób użycia: greplog {słowo_kluczowe} {nazwa_dziennika}
#
# błędy: nie sprawdza poprawności liczby argumentów

# buduj nazwę raportu ze słowa kluczowego i daty
log_report=/tmp/$1.logreport.`date '+%d%m%y'`

# buduj nagłówek raportu, zawierający typ systemu, nazwę hosta,
# datę i czas
echo "=====================================================" \
    > $log_report
echo "      DZIENNIK ZDARZEŃ SYSTEMOWYCH" >> $log_report
uname -a >> $log_report
echo "Raport dziennika dla " `hostname -f` "z dnia " `date '+%c'` \
    >> $log_report
echo "=====================================================" \
    >> $log_report; echo "" >> $log_report
```

```
# rozpoczęcie przeszukiwania dziennika
echo "Szukane hasło: " $1 "od dnia:" `date '+%r'` >> $log_report
echo "" >> $log_report

# pobranie i zapisanie wyników szukania słowa kluczowego ($1)
# w pliku dziennika ($2)
grep -i $1 $2 >> log_report

# stopka raportu z datą
echo "" >> $log_report
echo "Koniec pracy " >> $log_report `date '+%r'` >> $log_report

# wyślij raport do użytkownika root
mail -s "Analiza dziennika $2" root < $log_report

# usuń plik raportu i zakończ pracę
rm $log_report
exit 0
```

Powyższy skrypt tworzy zmienną `$log_report`, w której zapisuje nazwę pliku generowanego raportu. Słowo kluczowe `$1` i pierwszy argument w wierszu poleceń służą do zbudowania nazwy pliku wraz z bieżącą datą. Nagłówek raportu zawiera kilka wierszy sformatowanego tekstu, nazwę systemu (wziętą z wyjścia polecenia `uname`), nazwę hosta i datę. Skrypt rejestruje początek wyszukiwania, a wszystkie dopasowane słowa kluczowe z pliku dziennika zostają dodane do raportu. Do raportu zostaje dodana stopka zawierająca nazwę i datę raportu, który następnie zostaje wysłany do użytkownika *root* wiadomością e-mail z wyszukiwanym terminem zapisanym jako tytuł wiadomości, a plik *$log_report* zostaje usunięty.

Aby przetestować skrypt, upewnij się, że w systemie uruchomiony jest demon `syslogd`. Podaj słowo kluczowe i ścieżkę do dziennika zdarzeń systemowych */var/log/messages*, na przykład:

```
greplog FAILED /var/log/messages
```

Jeśli w systemie wystąpiły jakieś błędy logowania, użytkownik *root* powinien otrzymać wiadomość e-mail. Jeżeli usługi poczty elektronicznej zostały poprawnie skonfigurowane (zobacz „Konfiguracja konta pocztowego w SUSE Linux” w rozdziale 13.), wiadomość powinna dotrzeć do superużytkownika.

Zmienne wbudowane

Jądro Linuksa udostępnia powłoce kilka zmiennych wbudowanych, których wartości użytkownik nie może zmienić. Należą do nich:

$# — liczba parametrów pozycyjnych przekazanych do powłoki (tej zmiennej użyliśmy w poprzednim podrozdziale).

$? — kod zakończenia ostatniego polecenia lub programu powłoki wykonanego w powłoce (zwrócona wartość).

`$0` — nazwa programu powłoki.

`$*` — pojedynczy łańcuch wszystkich argumentów przekazanych w chwili wywołania programu powłoki.

To tylko kilka podstawowych zadań i rzeczy, które można zrobić z powłoką. Zagłębimy się w ten temat w rozdziale 33., „Wiersz polecenia — kurs zaawansowany".

Odnośniki

- *http://www.cryptonomicon.com/beginning.html* — esej Neala Stephensona *In the Beginning Was the Command Line*. Autoryzowana „aktualizacja" tego eseju, autorstwa Garretta Birkela, *The Command Line in 2004* dostępna jest pod adresem *http://garote.bdmonkeys.net/commandline/*.
- *http://www.gnu.org/software/bash/* — oficjalna witryna bash w GNU Project. Można tu pobrać najnowszą wersję oprogramowania i przeczytać dokument *Introduction to Bash*.
- *http://cnswww.cns.cwru.edu/~chet/bash/bashtop.html* — strona Cheta Rameya, oficjalnego opiekuna projektu bash. Bardzo przypomina stronę oficjalną, lecz dodatkowo zawiera listę FAQ.
- *http://www.tldp.org/LDP/Bash-Beginners-Guide/html/* — podręcznik bash dla początkujących.
- *http://ldp.nllgg.nl/LDP/abs/html* — zaawansowany poradnik pisania skryptów powłoki bash. „Ten podręcznik nie zakłada uprzedniego doświadczenia w pisaniu skryptów lub programowaniu, lecz szybko przechodzi do średnio zaawansowanego lub zaawansowanego poziomu instrukcji". Tekst można czytać online oraz pobrać jako PDF lub źródła SGML.
- *http://www.filibeto.org/sun/lib/development/shell/config_the_bash_shell.html* — konfiguracja powłoki bash.
- *http://www.justlinux.com/nhf/Shells/Basic_Console_Commands.html* — nauczenie się tych poleceń pozwoli Czytelnikowi mówić, że zna bash „wystarczająco dobrze".
- *http://www.kornshell.com/doc* — różnorodna dokumentacja komercyjnej powłoki Korn, ksh.
- *http://web.cs.mun.ca/~michael/pdksh* — serwis wersji ogólnie dostępnej (ang. *public domain*) powłoki Korn. Ta wersja zawarta jest w systemie SUSE Linux.
- *http://www.tcsh.org* — Wiki tcsh. Wprawdzie nie zawiera zbyt bogatych informacji, lecz warto zapoznać się z FAQ i stroną TipsNTricks. Dostępna jest też subskrypcja dwóch list dystrybucyjnych.
- *http://www.faqs.org/faqs/unix-faq/shell/csh-whynot* — sławny esej guru języka Perl Toma Christiansena *Csh Programming Considered Harmful*. Wyjaśnia, dlaczego pisanie skryptów w powłoce C nie jest najlepszym pomysłem.
- *http://www.zsh.org* — macierzysty serwis powłoki Z. Bardzo użyteczny.