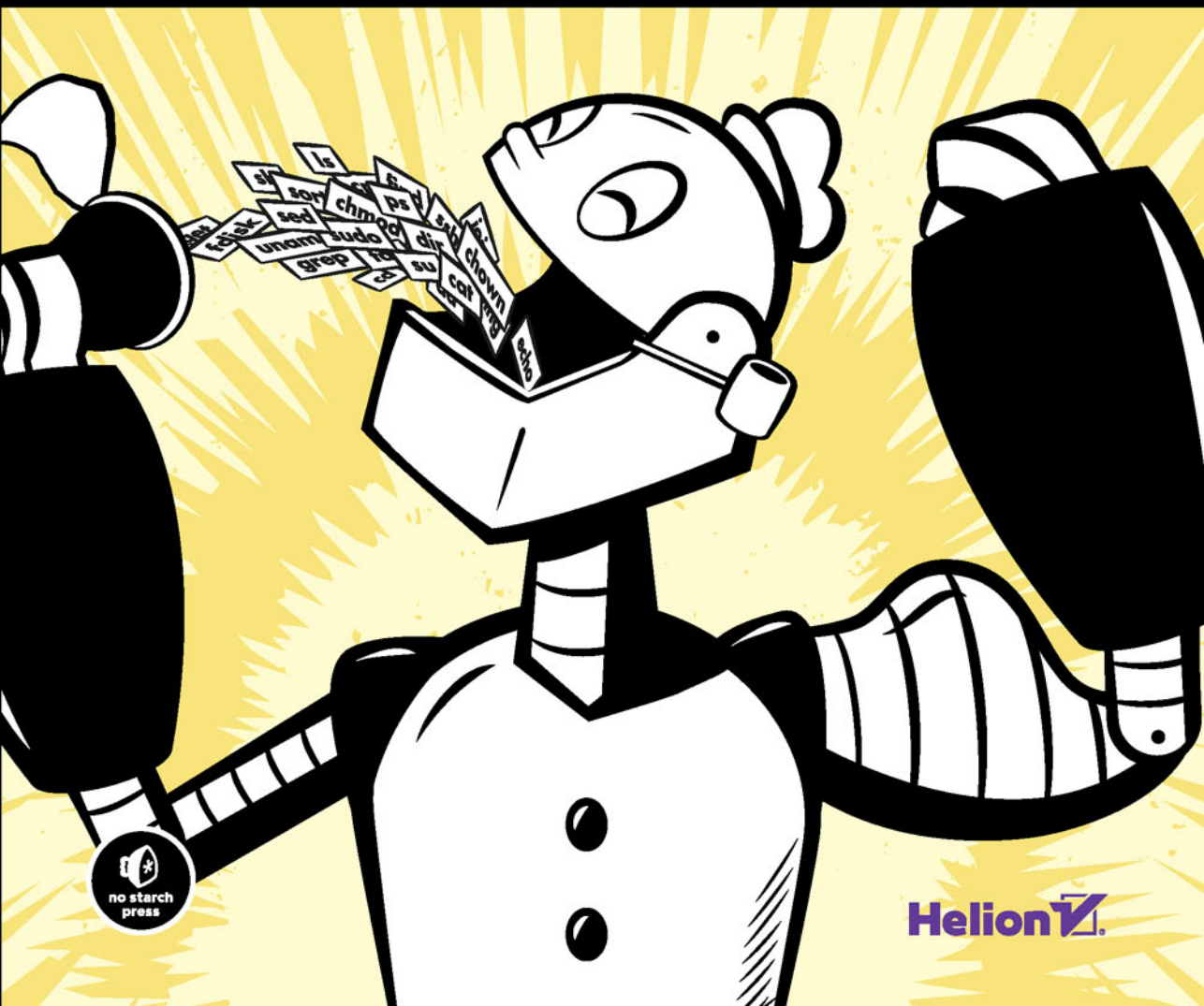


LINUX

WPROWADZENIE
DO WIERSZA POLECEŃ

WILLIAM E. SHOTTS JR



Helion

Tytuł oryginału: The Linux Command Line

Tłumaczenie: Joanna Zatorska (wstęp, rozdz. 3 – 36),
Przemysław Szeremiota (rozdz. 1 – 2)

ISBN: 978-83-283-0174-0

Original edition copyright © 2012 by William E. Shotts, Jr.
All rights reserved.

Published by arrangement with No Starch Press, Inc.

Polish edition copyright © 2015 by Helion SA.
All rights reserved.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Wydawnictwo HELION nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: helion@helion.pl
WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
<http://helion.pl/user/opinie/linwppw>
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

Spis treści

PODZIĘKOWANIA	19
WSTĘP	21

Część I Nauka powłoki

I

CZYM JEST POWŁOKA?	29
Emulatory terminali	29
Pierwsze kroki	30
Historia poleceń	30
Przesuwanie kursora	30
Pierwsze polecenia	31
Kończenie sesji terminala	32

2

NAWIGACJA	33
Hierarchia systemu plików	33
Bieżący katalog roboczy	34

Wypisywanie zawartości katalogu	35
Zmianie bieżącego katalogu roboczego	35
Ścieżki bezwzględne	35
Ścieżki względne	36
Przydatne skróty	37

3

PRZEGLĄD SYSTEMU 39

Więcej zabawy z ls	39
Opcje i argumenty	40
Długi format pod lupą	41
Sprawdzanie typu pliku za pomocą polecenia type	42
Wyświetlanie zawartości pliku za pomocą polecenia less	43
Wycieczka z przewodnikiem	44
Dowiązania symboliczne	45

4

MANIPULOWANIE PLIKAMI I KATALOGAMI 49

Wieloznaczniki	50
mkdir — tworzenie katalogów	52
cp — kopiowanie plików i katalogów	52
mv — przenoszenie plików i zmiana ich nazw	53
rm — usuwanie plików i katalogów	54
ln — tworzenie dowiązań	55
Dowiązania twarde	56
Dowiązania symboliczne	56
Zbudujmy plac zabaw	57
Tworzenie katalogów	57
Kopiowanie plików	57
Przenoszenie plików i zmiana ich nazw	58
Tworzenie dowiązań twarde	59
Tworzenie dowiązań symbolicznych	60
Usuwanie plików i katalogów	61
Uwagi końcowe	63

5

POLECENIA 65

Czym właściwie są polecenia?	65
Identyfikowanie poleceń	66
type — wyświetlanie typu polecenia	66
which — wyświetlanie lokalizacji pliku wykonywalnego	66
Pobieranie dokumentacji polecenia	67
help — uzyskiwanie pomocy dla poleceń wbudowanych w powłokę	67
--help — wyświetlanie informacji o użyciu	68
man — wyświetlanie podręcznika programu	68
apropos — wyświetlanie odpowiednich poleceń	69

whatis — wyświetlanie bardzo krótkiego opisu polecenia	70
info — wyświetlanie informacji o programie	70
README i inne pliki dokumentacji programu	72
Tworzenie własnych poleceń z wykorzystaniem polecenia alias	72
Powrót do starych przyjaciół	74
6	
PRZEKIEROWANIA	75
Standardowy strumień wejścia, wyjścia oraz błędów	76
Przekierowanie standardowego strumienia wyjścia	76
Przekierowanie standardowego strumienia błędów	78
Przekierowanie standardowego strumienia wyjścia i standardowego strumienia błędów do jednego pliku	78
Usuwanie niepotrzebnych danych wynikowych	79
Przekierowanie standardowego strumienia wejścia	80
Potoki	82
Filtry	82
uniq — zgłaszanie lub pomijanie powtarzających się wierszy	83
wc — wypisywanie liczników wierszy, słów oraz bajtów	83
grep — wypisywanie wierszy pasujących do wzorca	83
head (tail) — zwracanie początku (końca) pliku	84
tee — pobieranie danych ze standardowego strumienia wejścia, przekazywanie ich do standardowego strumienia wyjścia i do plików	85
Uwagi końcowe	86
7	
SPOJRZENIE NA ŚWIAT Z PUNKTU WIDZENIA POWŁOKI	89
Interpretacja poleceń	89
Interpretacja ścieżek	90
Interpretacja tyldy	91
Interpretacja wyrażeń arytmetycznych	92
Interpretacja nawiasów	93
Interpretacja parametrów	94
Podstawianie wyników poleceń	95
Cytowanie	96
Cudzysłowy podwójne	96
Pojedyncze cudzysłowy	98
Interpretowanie znaków	99
Uwagi końcowe	100
8	
ZAAWANSOWANE SZTUCZKI ZWIĄZANE Z KLAWIATURĄ	101
Edytowanie wiersza poleceń	102
Przemieszczanie kursora	102
Modyfikowanie tekstu	102
Wycinanie i wklejanie tekstu	103

Uzupełnianie	104
Korzystanie z historii	105
Przeszukiwanie historii	106
Interpretacja historii	107
Uwagi końcowe	108

9

UPRAWNIENIA 109

Właściciele, członkowie grupy i wszyscy pozostali	110
Odczyt, zapis i wykonywanie	111
chmod — zmiana trybu pliku	113
Ustawianie trybu pliku z poziomu interfejsu graficznego	116
umask — ustawianie uprawnień domyślnych	116
Zmiana tożsamości	119
su — uruchamianie powłoki z identyfikatorem zastępczego użytkownika i grupy	120
sudo — wykonywanie polecenia jako inny użytkownik	121
chown — zmiana właściciela pliku i grupy	122
chgrp — zmiana przypisania do grupy	123
Ćwiczenia dotyczące własnych uprawnień	124
Zmiana własnego hasła	126

10

PROCESY 127

Jak działa proces?	128
Wyświetlanie procesów za pomocą polecenia ps	128
Dynamiczne wyświetlanie procesów za pomocą polecenia top	130
Sterowanie procesami	131
Zatrzymywanie procesu	132
Umieszczanie procesu w tle	133
Przywracanie procesu do pierwszego planu	134
Zatrzymywanie (wstrzymywanie) procesu	134
Sygnaly	135
Wysyłanie sygnałów do procesów za pomocą polecenia kill	135
Wysyłanie sygnałów do wielu procesów za pomocą polecenia killall	137
Więcej poleceń dotyczących procesów	138

Część II

Konfiguracja i środowisko

11

ŚRODOWISKO 141

Co jest przechowywane w środowisku?	141
Przeglądanie środowiska	142
Niektóre ciekawe zmienne	143

W jaki sposób konfigurowane jest środowisko?	143
Powłoki logowania i niesłużące do logowania	144
Czym jest plik startowy?	145
Modyfikowanie środowiska	147
Które pliki należy zmodyfikować?	147
Edytory tekstu	147
Korzystanie z edytora tekstu	148
Aktywowanie naszych zmian	150
Uwagi końcowe	151

12

ŁAGODNE WPROWADZENIE DO VI 153

Dlaczego należy się nauczyć vi?	153
Krótkie wprowadzenie	154
Uruchamianie i zatrzymywanie vi	154
Tryby edycji	155
Włączanie trybu edycji	156
Zapisywanie pracy	157
Zmiana położenia kursora	158
Podstawowa edycja	159
Dodawanie tekstu	159
Otwieranie wiersza	160
Usuwanie tekstu	160
Wycinanie, kopiowanie i wklejanie tekstu	162
Łączenie wierszy	163
Szukanie i zastępowanie	163
Przeszukiwanie wiersza	163
Przeszukiwanie całego pliku	164
Wyszukiwanie i zastępowanie globalne	164
Edycja wielu plików	166
Przełączanie między plikami	166
Otwieranie do edycji dodatkowych plików	167
Kopiowanie treści z jednego pliku do drugiego	168
Wstawianie treści całego pliku do drugiego pliku	169
Zapisywanie zmian	169

13

DOSTOSOWYWANIE ZNAKU ZACHĘTY 171

Anatomia znaku zachęty	171
Alternatywne projekty znaków zachęty	172
Dodawanie koloru	174
Przesuwanie kursora	176
Zapisywanie znaku zachęty	177
Uwagi końcowe	177

Część III

Popularne zadania i podstawowe narzędzia

14

ZARZĄDZANIE PAKIETAMI	181
Systemy zarządzania pakietami	182
Jak działają systemy zarządzania pakietami?	182
Pliki pakietu	182
Repozytoria	183
Zależności	183
Narzędzia zarządzania pakietami wysokiego i niskiego poziomu	184
Popularne zadania zarządzania pakietami	184
Szukanie pakietu w repozytorium	184
Instalowanie pakietu z repozytorium	185
Instalowanie pakietu z wykorzystaniem pliku pakietu	185
Usuwanie pakietu	186
Uaktualnianie pakietów z repozytorium	186
Uaktualnianie pakietów za pomocą pliku pakietu	186
Wyświetlanie zainstalowanych pakietów	187
Sprawdzanie, czy pakiet jest zainstalowany	187
Wyświetlanie informacji o zainstalowanym pakiecie	188
Sprawdzanie, który pakiet zainstalował plik	188
Uwagi końcowe	188

15

NOŚNIKI DANYCH	191
Montowanie i odmontowywanie urządzeń pamięciowych	192
Wyświetlanie listy zamontowanych systemów plików	193
Ustalanie nazwy urządzenia	196
Tworzenie nowych systemów plików	199
Manipulowanie partycjami z wykorzystaniem fdisk	200
Tworzenie nowego systemu plików z wykorzystaniem mkfs	202
Testowanie i naprawa systemów plików	203
Formatowanie dyskietek	204
Przenoszenie danych bezpośrednio do urządzeń oraz z urządzeń	204
Tworzenie obrazów dysków CD	205
Tworzenie obrazu kopii dysku CD	205
Tworzenie obrazu na podstawie zbioru plików	206
Zapisywanie obrazów CD	206
Bezpośrednie montowanie obrazu ISO	206
Opróżnianie zapisywalnego dysku CD	207
Zapisywanie obrazu	207
Dodatkowe informacje	207

16

ZAGADNIENIA SIECIOWE	209
Sprawdzanie i monitorowanie sieci	210
ping — wysyłanie pakietu specjalnego do hosta sieciowego	210
traceroute — śledzenie trasy pakietu sieciowego	211
netstat — sprawdzanie ustawień sieci i statystyk	212
Przenoszenie plików poprzez sieć	213
ftp — transfer plików z wykorzystaniem protokołu transferu plików	214
lftp — ulepszony ftp	215
wget — nieinteraktywny program do pobierania plików z sieci	216
Bezpieczna komunikacja z hostami zdalnymi	216
ssh — bezpieczne logowanie do komputerów zdalnych	216
scp i sftp — bezpieczny transfer plików	220

17

SZUKANIE PLIKÓW	223
locate — łatwy sposób szukania plików	224
find — trudny sposób wyszukiwania plików	225
Testy	226
Operatory	227
Akcje	230
Powrót do placu zabaw	234
Opcje	237

18

ARCHIWIZACJA I KOPIE ZAPASOWE	239
Kompresowanie plików	240
gzip — kompresowanie i wyodrębnianie plików	240
bzip2 — wyższy poziom kompresji kosztem szybkości	242
Archiwizacja plików	243
tar — narzędzie do archiwizacji taśmowej	243
zip — pakowanie i kompresowanie plików	248
Synchronizacja plików i katalogów	251
rsync — synchronizacja zdalnych plików i katalogów	251
Korzystanie z polecenia rsync poprzez sieć	253

19

WYRAŻENIA REGULARNE	255
Co to są wyrażenia regularne?	255
grep — wyszukiwanie w tekście	256
Metaznaki i literały	258
Znak dowolny	258
Kotwice	259

Wyrażenia w nawiasach i klasy znaków	260
Zaprzeczenie	260
Tradycyjne zakresy znaków	261
Klasy znaków POSIX	262
Podstawowy POSIX a rozszerzone wyrażenia regularne	264
Alternatywa	266
Kwantyfikatory	267
? — dopasowuje element zero lub jeden raz	267
* — dopasowuje element zero lub więcej razy	268
+ — dopasowuje element raz lub więcej razy	269
{} — dopasowuje element określoną liczbę razy	269
Zapręgamy wyrażenia regularne do pracy	270
Sprawdzanie listy telefonicznej za pomocą polecenia grep	270
Szukanie brzydkich nazw plików z wykorzystaniem polecenia find	271
Wyszukiwanie plików za pomocą polecenia locate	272
Wyszukiwanie tekstu za pomocą programów less i vim	272
Uwagi końcowe	274

20

PRZETWARZANIE TEKSTU 275

Zastosowanie tekstu	276
Dokumenty	276
Strony WWW	276
E-mail	276
Wyjście drukarki	276
Kod źródłowy programów	277
Ponowne odwiedziny u starych przyjaciół	277
cat — łączenie plików i wypisywanie ich zawartości w standardowym strumieniu wyjścia	277
sort — sortowanie wierszy plików tekstowych	279
uniq — zgłaszanie lub pomijanie powtarzających się wierszy	285
Cięcie i krojenie	287
cut — usuwanie fragmentów z każdego wiersza plików	287
paste — łączenie wierszy w pliku	290
join — łączenie dwóch plików na podstawie wspólnego pola	291
Porównywanie tekstu	293
comm — porównywanie dwóch posortowanych plików wiersz po wierszu	293
diff — porównywanie plików wiersz po wierszu	294
patch — dołączanie do oryginału pliku z różnicami	296
Edycja w locie	298
tr — transliterowanie lub usuwanie znaków	298
sed — edytor strumieniowy służący do filtrowania i przekształcania tekstu	300
aspell — interaktywny program do sprawdzania pisowni	307
Uwagi końcowe	310
Dodatkowe informacje	311

21

FORMATOWANIE WYNIKÓW 313

Proste narzędzia formatowania	313
nl — wstawianie numerów wierszy	314
fold — zawijanie każdego wiersza do określonej długości	317
fmt — prosty program do formatowania tekstu	317
pr — formatowanie tekstu do druku	320
printf — formatowanie i wypisywanie danych	321
Systemy formatowania dokumentów	324
Rodzina roff i T _E X	324
groff — system formatowania dokumentów	325
Uwagi końcowe	329

22

DRUKOWANIE 331

Krótką historia druku	332
Drukowanie w zamierzonych czasach	332
Drukarki oparte na znakach	332
Drukarki graficzne	333
Drukowanie w systemie Linux	334
Przygotowanie plików do druku	335
pr — przekształcanie plików tekstowych przeznaczonych do druku	335
Przesyłanie zadania drukowania do drukarki	335
lpr — drukowanie plików (styl Berkeley)	336
lp — drukowanie plików (styl Systemu V)	337
Inna opcja — a2ps	337
Monitorowanie zadań drukowania i sterowanie nimi	339
lpstat — wyświetlanie informacji o stanie drukarki	341
lpq — wyświetlanie statusu kolejki drukarki	342
lprm i cancel — anulowanie zadań drukowania	342

23

KOMPILOWANIE PROGRAMÓW 343

Czym jest kompilowanie?	344
Czy wszystkie programy są skompilowane?	345
Kompilowanie programu w języku C	346
Uzyskiwanie kodu źródłowego	346
Sprawdzanie zawartości drzewa źródłowego	348
Budowanie programu	349
Instalowanie programu	353
Uwagi końcowe	353

Część IV

Tworzenie skryptów powłoki

24

PISANIE PIERWSZEGO SKRYPTU 357

Czym są skrypty powłoki?	357
Jak napisać skrypt powłoki?	358
Format pliku skryptu	358
Uprawnienia do wykonywania	359
Lokalizacja pliku skryptu	359
Dobre lokalizacje dla skryptów	361
Więcej trików formatowania	361
Długie nazwy opcji	361
Wcięcia i kontynuacja wierszy	362
Uwagi końcowe	363

25

ROZPOCZYNANIE PROJEKTU 365

Pierwszy etap — minimalny dokument	365
Drugi etap — dodawanie pewnych danych	368
Zmienne i stałe	368
Tworzenie zmiennych i stałych	369
Przypisywanie wartości do zmiennych i stałych	371
Dokumenty włączone	372
Uwagi końcowe	375

26

PROJEKTOWANIE ZSTĘPUJĄCE 377

Funkcje powłoki	378
Zmienne lokalne	381
Utrzymywanie działania skryptów	382
Uwagi końcowe	385

27

STEROWANIE PRZEPLYWEM — ROZGAŁĘZIENIA IF 387

Wykorzystanie if	388
Status wyjścia	388
Korzystanie z testu	390
Funkcje plikowe	390
Funkcje tekstowe	392
Funkcje liczbowe	394
Nowocześniejsza wersja programu test	395
(()) — przeznaczone dla liczb całkowitych	396

Łączenie wyrażień	397
Operatory sterowania — inny sposób rozgałęziania	400
Uwagi końcowe	401
28	
ODCZYT WEJŚCIA Z KLAWIATURY	403
read — odczyt danych ze standardowego strumienia wejścia	404
Opcje	407
Rozdzielanie pól wejściowych za pomocą IFS	407
Weryfikacja wejścia	410
Menu	411
Uwagi końcowe	412
Dodatkowe informacje	413
29	
STEROWANIE PRZEPLYWEM — PĘTLE WHILE I UNTIL	415
Pętle	416
while	416
Ucieczka z pętli	418
until	420
Odczyt plików za pomocą pętli	420
Uwagi końcowe	421
30	
USUWANIE BŁĘDÓW	423
Błędy składniowe	423
Brakujące cudzysłowy	424
Brakujące lub niespodziewane tokeny	425
Nieprzewidziane interpretacje	425
Błędy logiczne	427
Programowanie defensywne	427
Weryfikacja wejścia	428
Testowanie	429
Elementy zastępcze	429
Przypadki testowe	430
Debugowanie	431
Znalezienie miejsca problemu	431
Śledzenie	432
Sprawdzanie wartości podczas wykonywania	434
Uwagi końcowe	434

31	
STEROWANIE PRZEPŁYWEM — ROZGAŁĘZIENIA CASE	435
case	435
Wzorce	437
Łączenie wielu wzorców	438
Uwagi końcowe	439
32	
PARAMETRY POZYCYJNE	441
Dostęp do wiersza poleceń	441
Ustalanie liczby argumentów	442
shift — uzyskiwanie dostępu do wielu argumentów	443
Proste programy	444
Korzystanie z parametrów pozycyjnych wraz z funkcjami powłoki	445
Masowa obsługa parametrów pozycyjnych	446
Bardziej kompletne programy	448
Uwagi końcowe	451
33	
STEROWANIE PRZEPŁYWEM — PĘTLA FOR	455
for — tradycyjna forma powłoki	455
for — forma języka C	458
Uwagi końcowe	459
34	
ŁAŃCUCHY TEKSTOWE I LICZBY	461
Interpretacja parametrów	461
Podstawowe parametry	462
Interpretacje służące do zarządzania pustymi zmiennymi	462
Interpretacje, które zwracają nazwy zmiennych	464
Operacje na łańcuchach tekstowych	464
Interpretacja wyrażeń arytmetycznych	467
Liczby o różnej podstawie	467
Operatory jednoargumentowe	468
Prosta arytmetyka	468
Przypisanie	469
Operacje bitowe	472
Logika	472
bc — język kalkulatora dowolnej precyzji	475
Korzystanie z bc	475
Przykładowy skrypt	476
Uwagi końcowe	477
Dodatkowe informacje	477

35

TABLICE	479
Czym są tablice?	479
Tworzenie tablic	480
Przypisywanie wartości do tablicy	480
Dostęp do elementów tablicy	481
Operacje na tablicach	483
Wyświetlanie zawartości całej tablicy	483
Określanie liczby elementów tablicy	484
Znajdowanie indeksów wykorzystanych przez tablicę	484
Dodawanie elementów na końcu tablicy	485
Sortowanie tablicy	485
Usuwanie tablicy	485
Uwagi końcowe	486

36

EGZOTYKA	489
Polecenia grupowe i podpowtoki	489
Wykonywanie przekierowań	490
Substytucja procesu	490
Pułapki	493
Wykonywanie asynchroniczne	496
wait	496
Potoki nazwane	497
Ustawianie potoku nazwanego	498
Korzystanie z potoków nazwanych	498
Uwagi końcowe	499

SKOROWIDZ	501
------------------------	------------

10

Procesy

WSPÓLCZESNE SYSTEMY OPERACYJNE SĄ ZWYKLE *WIELOZADANIOWE*, CO OZNACZA, ŻE POPRZEZ SZYBKĄ ZMIANĘ PROGRAMU, KTÓRY JEST WYKONYWANY, SPRAWIAJĄ WRAŻENIE WYKONYWANIA KILKU ZADAŃ JEDNOCZEŚNIE. JĄDRO SYSTEMU LINUX zarządza tym poprzez wykorzystanie *procesów*. Procesy są sposobem organizowania przez Linux różnych programów oczekujących na swoją kolej do zasobów CPU.

Niekiedy komputer staje się powolny lub aplikacja przestaje odpowiadać. W tym rozdziale przyjrzymy się niektórym narzędziom dostępnym w wierszu poleceń, które umożliwiają nam sprawdzenie, co wykonują programy i jak zakończyć procesy, które przebiegają niewłaściwie.

W tym rozdziale zostaną wprowadzone następujące polecenia:

- `ps` — wyświetla listę bieżących procesów,
- `top` — wyświetla zadania,
- `jobs` — wypisuje aktywne zadania,
- `bg` — umieszcza zadanie w tle,
- `fg` — umieszcza zadanie na pierwszym planie,
- `kill` — wysyła sygnał do procesu,
- `killall` — kończy proces o podanej nazwie,
- `shutdown` — zamyka lub ponownie uruchamia system.

Jak działa proces?

Podczas rozruchu systemu jądro inicjalizuje kilka własnych zadań w postaci procesów i uruchamia program zwany `init`. Z kolei `init` uruchamia serię skryptów powłoki (znajdujących się w katalogu `/etc`) zwanych *skryptami inicjalizacyjnymi*, które uruchamiają wszystkie usługi systemowe. Wiele z tych usług jest zaimplementowanych w postaci demonów, czyli programów, które wykonują swoje zadania w tle i nie posiadają żadnego interfejsu użytkownika. Dlatego nawet jeśli nie jesteśmy zalogowani, system jest zawsze choć trochę zajęty wykonywaniem rutynowych zadań.

Możliwość uruchamiania programów przez inne programy jest wyrażana na drzewie procesów w postaci *procesu macierzystego* tworzącego *proces potomny*.

Jądro ułatwia sobie organizację, utrzymując informacje o każdym procesie. Na przykład do każdego procesu przypisywany jest numer zwany *identyfikatorem procesu* (*PID* — ang. *process id*). Numery PID są przydzielane w kolejności rosnącej, przy czym `init` zawsze otrzymuje numer 1. Jądro kontroluje także pamięć przydzieloną do każdego procesu, a także gotowość procesów do wznowienia działania. Podobnie jak pliki, procesy również posiadają właścicieli, identyfikatory użytkownika, identyfikatory EUID itd.

Wyświetlanie procesów za pomocą polecenia `ps`

Najczęściej używanym poleceniem służącym do wyświetlania procesów (jest ich kilka) jest `ps`. Program `ps` przyjmuje wiele opcji, jednak w najprostszy sposób używa się go następująco:

```
[me@linuxbox ~]$ ps
  PID TTY          TIME CMD
 5198 pts/1    00:00:00 bash
10129 pts/1    00:00:00 ps
```

W wyniku działania powyższego polecenia uzyskaliśmy listę dwóch procesów: 5198 i 10129, które odpowiadają programom `bash` i `ps`. Jak widać, domyślnie polecenie `ps` nie wyświetla zbyt wiele, tylko procesy związane z bieżącą sesją terminala. Aby zobaczyć więcej, musimy dodać kilka opcji. Zanim to zrobimy, spójrzmy na inne pola wyświetlone przez `ps`. `TTY` jest skrótem od słowa *teletype* i odnosi się do *terminala kontrolnego* procesu. Jest to pozostałość po dawnych czasach systemu Unix. Pole `TIME` oznacza czas wykorzystania CPU przez proces. Jak widać, żaden z procesów nie wymagał od komputera dużego nakładu pracy.

Po dodaniu opcji możemy uzyskać lepszy obraz działania systemu:

```
[me@linuxbox ~]$ ps x
  PID TTY          STAT TIME COMMAND
 2799 ?        Ss1   0:00 /usr/libexec/bonobo-activation-server -ac
 2820 ?        S1    0:01 /usr/libexec/evolution-data-server-1.10 --
15647 ?        Ss    0:00 /bin/sh /usr/bin/startkde
```

```

15751 ?      Ss   0:00 /usr/bin/ssh-agent /usr/bin/dbus-launch --
15754 ?      S    0:00 /usr/bin/dbus-launch --exit-with-session
15755 ?      Ss   0:01 /bin/dbus-daemon --fork --print-pid 4 --pr
15774 ?      Ss   0:02 /usr/bin/gpg-agent -s --daemon
15793 ?      S    0:00 start_kdeinit --new-startup +kcmnit_start
15794 ?      Ss   0:00 kdeinit Running...
15797 ?      S    0:00 dcopserver --nosid
i wiele więcej...

```

Dodanie opcji `x` (zauważmy brak myślnika przed opcją), informuje `ps` o konieczności wyświetlenia informacji o wszystkich procesach, niezależnie od terminala, pod którego kontrolą pozostają (o ile taki istnieje). Obecność znaku `?` w kolumnie TTY oznacza brak terminala kontrolnego. Korzystając z tej opcji, możemy się dowiedzieć, które procesy należą do nas.

Ponieważ w systemie działa wiele procesów, `ps` wyświetla długą listę. Zwykle warto przekazać potokiem wynik polecenia do polecenia `less`, aby ułatwić przeglądanie. Niektóre kombinacje opcji tworzą też długie wiersze tekstu, dlatego dobrym pomysłem będzie zmaksymalizowanie okna emulatora terminala.

Na liście wynikowej pojawiła się nowa kolumna `STAT`. Nazwa ta jest skrótem od angielskiego wyrazu *state* („stan”) i zawiera bieżący stan procesu, zgodnie z opisem w tabeli 10.1.

Tabela 10.1. Stany procesu

Status	Znaczenie
R	Uruchomiony. Proces jest uruchomiony lub gotowy do uruchomienia.
S	Uśpiony. Proces nie działa; raczej oczekuje na zdarzenie, takie jak naciśnięcie klawisza, lub na otrzymanie pakietu sieciowego.
D	Uśpiony nieprzerwalny. Proces oczekuje na urządzenie wejścia-wyjścia, takie jak dysk.
T	Zatrzymany. Proces, który otrzymał instrukcję zatrzymania (więcej informacji na ten temat znajdziemy w dalszej części rozdziału).
Z	Proces działający nieprawidłowo lub proces „zombie”. Jest to proces potomny, który zakończył działanie, jednak nie został usunięty przez proces macierzysty.
<	Proces o wysokim priorytecie. Procesowi możemy nadać większe znaczenie, przydzielając mu więcej czasu na CPU. Ta właściwość procesu nosi nazwę <i>niceness</i> . Możemy powiedzieć, że proces o wysokim priorytecie jest gorszy, ponieważ potrzebuje więcej czasu CPU, co pozostawia mniej czasu dla pozostałych.
N	Proces o niskim priorytecie. Proces taki (lepszy proces) uzyska dostęp do procesora tylko wtedy, gdy zostaną obsłużone pozostałe procesy, mające wyższy priorytet.

Po znaku stanu procesu mogą wystąpić inne znaki. Oznaczają one różne egzotyczne cechy procesu. Szczegółowe informacje na ten temat można uzyskać w podręczniku `man` polecenia `ps`.

Inny popularny zestaw opcji to `aux` (bez myślnika przed opcją). Pozwala uzyskać jeszcze więcej informacji:

```
[me@linuxbox ~]$ ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0  2136  644 ?        Ss   Mar05   0:31 init
root         2  0.0  0.0    0     0 ?        S<   Mar05   0:00 [kt]
root         3  0.0  0.0    0     0 ?        S<   Mar05   0:00 [mi]
root         4  0.0  0.0    0     0 ?        S<   Mar05   0:00 [ks]
root         5  0.0  0.0    0     0 ?        S<   Mar05   0:06 [wa]
root         6  0.0  0.0    0     0 ?        S<   Mar05   0:36 [ev]
root         7  0.0  0.0    0     0 ?        S<   Mar05   0:00 [kh]
i wiele więcej...
```

Ten zestaw opcji wyświetla procesy należące do każdego użytkownika. Użycie tych opcji bez myślnika sprawia, że polecenie będzie działać w „stylu BDS”. Wersja programu ps dostępna w systemie Linux może emulować zachowania programu ps z kilku dystrybucji Uniksa. Opcje te wyświetlają dodatkowe kolumny, opisane w tabeli 10.2.

Tabela 10.2.

Nagłówek	Znaczenie
USER	Identyfikator użytkownika. Jest to właściciel procesu.
%CPU	Wykorzystanie CPU w procentach.
%MEM	Wykorzystanie pamięci w procentach.
VSZ	Rozmiar pamięci wirtualnej.
RSS	Skrót od <i>Resident Set Size</i> . Ilość pamięci fizycznej (RAM) wykorzystywanej przez proces w kilobajtach.
START	Czas rozpoczęcia procesu. Wartości powyżej 24 godzin reprezentowane są przez daty.

Dynamiczne wyświetlanie procesów za pomocą polecenia top

Chociaż polecenie ps umożliwia wyświetlenie wielu informacji o działaniu maszyny, to pozwala uzyskać jedynie obraz stanu maszyny w momencie wykonania polecenia ps. Aby wyświetlić bardziej dynamiczny obraz aktywności maszyny, korzystamy z polecenia top:

```
[me@linuxbox ~]$ top
```

Program top wyświetla ciągle uaktualniany (domyślnie co 3 sekundy) obraz procesów w systemie, posortowanych według aktywności. Nazwa polecenia związana jest ze sposobem działania. Otóż program top wyświetla tylko procesy z górnej (ang. *top*) części listy procesów. Wynik działania polecenia top składa się z dwóch części: podsumowania stanu systemu w górnej części oraz tabeli procesów posortowanych według aktywności:

```
top - 14:59:20 up 6:30, 2 users, load average: 0.07, 0.02, 0.00
Tasks: 109 total, 1 running, 106 sleeping, 0 stopped, 2 zombie
Cpu(s): 0.7%us, 1.0%sy, 0.0%ni, 98.3%id, 0.0%wa, 0.0%hi, 0.0%si
Mem: 319496k total, 314860k used, 4636k free, 19392k buff
Swap: 875500k total, 149128k used, 726372k free, 114676k cach
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
6244	me	39	19	31752	3124	2188	S	6.3	1.0	16:24.42	trackerd
11071	me	20	0	2304	1092	840	R	1.3	0.3	0:00.14	top
6180	me	20	0	2700	1100	772	S	0.7	0.3	0:03.66	dbus-dae
6321	me	20	0	20944	7248	6560	S	0.7	2.3	2:51.38	multiloa
4955	root	20	0	104m	9668	5776	S	0.3	3.0	2:19.39	Xorg
1	root	20	0	2976	528	476	S	0.0	0.2	0:03.14	init
2	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	RT	-5	0	0	0	S	0.0	0.0	0:00.00	migratio
4	root	15	-5	0	0	0	S	0.0	0.0	0:00.72	ksoftirq
5	root	RT	-5	0	0	0	S	0.0	0.0	0:00.04	watchdog
6	root	15	-5	0	0	0	S	0.0	0.0	0:00.42	events/0
7	root	15	-5	0	0	0	S	0.0	0.0	0:00.06	khelper
41	root	15	-5	0	0	0	S	0.0	0.0	0:01.08	kblockd/
67	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	kseriod
114	root	20	0	0	0	0	S	0.0	0.0	0:01.62	pdflush
116	root	15	-5	0	0	0	S	0.0	0.0	0:02.44	kswapd0

Podsumowanie stanu systemu zawiera wiele przydatnych informacji, które zostały wyjaśnione w tabeli 10.3.

Program top przyjmuje zestaw poleceń z klawiatury. Dwa najciekawsze to h, które wyświetla ekran pomocy programu, oraz q, które zamyka program top.

Obydwa największe środowiska graficzne udostępniają aplikacje graficzne, które wyświetlają informacje podobne do top (w większości przypadków w podobny sposób, jak czyni to Windows), jednak uważam, że polecenie top jest lepsze niż wersje graficzne ze względu na szybkość i znacznie mniejsze wymagania co do zasobów systemu. W końcu nasz program monitorujący system nie powinien przyczyniać się do dalszego spowolnienia systemu, które przecież chcemy wyśledzić.

Sterowanie procesami

Wiemy już, jak wyświetlić informacje o procesach i jak je monitorować. Zobaczmy teraz, jak uzyskać nad nimi pewną kontrolę. Na potrzeby naszych ćwiczeń wykorzystamy mały program o nazwie xlogo. Program xlogo jest przykładowym programem dostarczanym z systemem X Window System (silnik odpowiadający za wyświetlanie grafiki na monitorze), który po prostu wyświetla okno o zmiennej wielkości zawierające logo X. Najpierw poznamy obiekt naszych eksperymentów:

```
[me@linuxbox ~]$ xlogo
```

Tabela 10.3. Pola informacyjne polecenia top

Rząd	Pole	Znaczenie
1	top	Nazwa programu.
	14:59:20	Bieżący czas.
	up 6:30	Wartość ta zwana jest <i>uptime</i> . Jest to czas, który upłynął od ostatniego uruchomienia systemu na maszynie. W tym przykładzie system został uruchomiony 6 i pół godziny temu.
	2 users	Zalogowanych jest dwóch użytkowników.
	load average:	Wartość <i>load average</i> oznacza liczbę procesów oczekujących na wznowienie; czyli jest to liczba procesów, które są uruchomione i dzielą między sobą czas CPU. Wyświetlone są trzy wartości, z których każda dotyczy różnego okresu. Pierwszą jest średnia dla ostatnich 60 sekund, kolejna dotyczy ostatnich 5 minut, a ostatnia dotyczy ostatnich 15 minut. Wartości poniżej 1,0 oznaczają, że maszyna nie jest zbyt zajęta.
2	Tasks:	Podsumowanie liczby procesów oraz ich różnych stanów.
	0.7%us	Procesy użytkownika wykorzystują 0,7% CPU. Dotyczy to procesów wykonywanych poza samym jądrem.
	1.0%sy	Procesy systemu (jądra) wykorzystują 1,0% CPU.
	0.0%ni	Procesy o niskim priorytecie wykorzystują 0,0% CPU.
	98.3%id	98,3% CPU jest niewykorzystane.
	0.0%wa	0,0% CPU oczekuje na urządzenie I/O.
4	Mem:	Pokazuje wykorzystanie fizycznej pamięci RAM.
5	Swap:	Pokazuje wykorzystanie przestrzeni wymiany (pamięci wirtualnej).

Gdy wpisujemy polecenie, gdzieś na ekranie pojawi się małe okno zawierające logo. W niektórych systemach xlogo może wyświetlić ostrzeżenie, które jednak możemy zignorować.

UWAGA *Jeśli w systemie nie zainstalowano programu xlogo, można zamiast niego wykorzystać program gedit lub kwrite.*

Zmieniając rozmiar okna, możemy sprawdzić, czy program xlogo pozostaje uruchomiony. Jeśli wielkość logo dopasuje się do nowego rozmiaru, oznacza to, że program działa.

Czy Czytelnik zauważył, że polecenie nie zwróciło wyniku w powłoce? Przyczyną tego zachowania jest fakt, że powłoka oczekuje na zakończenie działania programu, podobnie jak w przypadku pozostałych programów, z których dotychczas korzystaliśmy. Znak zachęty powróci do powłoki, gdy zamkniemy okno xlogo.

Zatrzymywanie procesu

Zobaczmy, co się stanie, gdy ponownie uruchomimy xlogo. Najpierw wpisujemy polecenie xlogo i sprawdzimy, czy program działa. Następnie powrócimy do okna terminala i użyjemy kombinacji klawiszy *Ctrl+C*.

```
[me@linuxbox ~]$ xlogo
[me@linuxbox ~]$
```

Zastosowanie kombinacji klawiszy *Ctrl+C* zatrzymuje program. Oznacza to, że grzecznie prosimy program o zakończenie działania. Po użyciu kombinacji klawiszy *Ctrl+C* okno *xlogo* zostało zamknięte, a do powłoki powrócił znak zachęty.

W ten sposób możemy zatrzymać wiele programów (lecz nie wszystkie).

Umieszczanie procesu w tle

Załóżmy, że chcemy, aby znak zachęty powrócił do powłoki, ale bez zatrzymywania programu *xlogo*. Wykonamy to, umieszczając proces *w tle*. Założmy, że terminal ma pierwszy plan (w którym widoczne są różne rzeczy, na przykład znak zachęty) oraz drugi plan, czyli tło (zawierający rzeczy ukryte pod powierzchnią). Aby uruchomić program i natychmiast umieścić go w tle, po poleceniu wpisujemy znak ampersand (&):

```
[me@linuxbox ~]$ xlogo &
[1] 28236
[me@linuxbox ~]$
```

Gdy wpisaliśmy polecenie, na ekranie pojawiło się okno *xlogo*, a do powłoki powrócił znak zachęty, jednak pojawiły się także pewne śmieszne cyferki. Ta informacja wchodzi w skład funkcji powłoki zwanej *sterowaniem zadaniami*. Powłoka informuje nas, że uruchomiła zadanie o numerze 1 ([1]) oraz że posiada ono PID 28236. Sprawdźmy nasz proces, wykonując polecenie *ps*:

```
[me@linuxbox ~]$ ps
  PID TTY          TIME CMD
 10603 pts/1    00:00:00 bash
 28236 pts/1    00:00:00 xlogo
 28239 pts/1    00:00:00 ps
```

Funkcja sterowania zadaniami powłoki umożliwia nam również wyświetlenie zadań uruchomionych z poziomu naszego terminala. Wykonując polecenie *jobs*, uzyskamy następującą listę:

```
[me@linuxbox ~]$ jobs
[1]+  Running                  xlogo &
```

Powyższy wynik informuje, że uruchomione jest jedno zadanie o numerze 1 oraz że polecenie miało postać *xlogo &*.

Przywracanie procesu do pierwszego planu

Proces działający w tle nie reaguje na wejście z klawiatury, ignoruje też próby przerwania procesu kombinacją klawiszy *Ctrl+C*. Aby przywrócić proces do pierwszego planu, korzystamy z polecenia *fg*, jak w poniższym przykładzie:

```
[me@linuxbox ~]$ jobs
[1]+  Running                  xlogo &
[me@linuxbox ~]$ fg %1
xlogo
```

Polecenie *fg*, po którym następuje znak procenta oraz numer zadania (zwany *jobspec*), załatwia sprawę. Jeśli w tle uruchomione jest tylko jedno polecenie, numer zadania jest opcjonalny. Aby zakończyć działanie programu *xlogo*, użyjmy kombinacji klawiszy *Ctrl+C*.

Zatrzymywanie (wstrzymywanie) procesu

Czasem chcemy zatrzymać działanie procesu bez jego zakończenia. Możliwość tę wykorzystuje się zwykle, aby można było przenieść proces pierwszoplanowy na drugi plan. Aby zatrzymać proces pierwszoplanowy, należy zastosować kombinację klawiszy *Ctrl+Z*. Spróbujmy. Wpiszmy *xlogo* w wierszu poleceń, naciśnijmy *Enter*, a następnie kombinację klawiszy *Ctrl+Z*.

```
[me@linuxbox ~]$ xlogo
[1]+  Stopped                  xlogo
[me@linuxbox ~]$
```

Po zatrzymaniu programu *xlogo* możemy zweryfikować, czy został zatrzymany, próbując zmienić rozmiar okna programu. Zauważymy, że wydaje się raczej martwe. Program możemy przywrócić na pierwszy plan, korzystając z polecenia *fg*, lub umieścić go w tle, używając polecenia *bg*:

```
[me@linuxbox ~]$ bg %1
[1]+  xlogo &
[me@linuxbox ~]$
```

Jeśli chodzi o polecenie *fg*, to jeśli istnieje tylko jedno zadanie, numer zadania jest opcjonalny.

Umieszczanie procesu pierwszoplanowego w tle jest przydatne, jeśli uruchamiamy program graficzny z wiersza poleceń, ale zapomnimy umieścić go w tle za pomocą znaku *&* wpisanego za poleceniem.

Dlaczego uruchamia się programy graficzne z wiersza poleceń? Istnieją dwa powody. Pierwszy wynika z tego, że program, który chcemy uruchomić, może nie być wymieniony w menu menedżera okna (na przykład *xlogo*).

Drugi wynika z tego, że program uruchomiony z wiersza poleceń będzie wyświetlać informacje o błędach, które mogłyby być niewidoczne w przypadku programu uruchomionego w interfejsie graficznym. Czasem program uruchomiony z poziomu menu graficznego nie zadziała. Jeśli wykorzystamy do tego celu wiersz poleceń, możemy zobaczyć komunikat o błędzie, informujący nas o problemie. Ponadto niektóre programy graficzne posiadają wiele ciekawych i przydatnych opcji wiersza poleceń.

Sygnaly

Polecenie `kill` jest wykorzystywane do „zabicia” (zakończenia) procesu. Pozwala nam to na zakończenie wykonywania programu, który działa niepoprawnie lub nie chce zakończyć działania samoczynnie. Poniżej przedstawiony jest przykład:

```
[me@linuxbox ~]$ xlogo &
[1] 28401
[me@linuxbox ~]$ kill 28401
[1]+  Terminated                  xlogo
```

Najpierw uruchamiamy w tle program `xlogo`. Powłoka wypisze numer zadania oraz PID procesu działającego w tle. Następnie używamy polecenia `kill` i podajemy PID procesu, który chcemy zakończyć. Zamiast numeru PID, moglibyśmy także użyć numeru zadania (na przykład `%1`).

Wygląda to na bardzo proste, ale to nie wszystko. Polecenie `kill` właściwie nie „zabija” procesów; raczej wysyła do nich sygnały. Sygnały stanowią jeden ze sposobów komunikacji systemu operacyjnego z programami. Działanie sygnałów poznaliśmy już przy korzystaniu z kombinacji klawiszy `Ctrl+C` i `Ctrl+Z`. Gdy użyjemy w terminalu jednej z tych kombinacji, terminal wyśle sygnał do programu pierwszoplanowego. W przypadku kombinacji `Ctrl+C` wysyłany jest sygnał `INT` (*Interrupt*); natomiast w przypadku kombinacji `Ctrl+Z` wysyłany jest sygnał `TSTP` (*Terminal Stop*). Z kolei programy „nasłuchują” sygnałów i mogą w odpowiedzi na nie wykonywać pewne działania. Możliwość nasłuchiwanie i działania w odpowiedzi na sygnały pozwala im na wykonywanie pewnych czynności, takich jak zapisywanie postępu pracy, gdy wysłany zostanie sygnał o zakończeniu.

Wysyłanie sygnałów do procesów za pomocą polecenia `kill`

Najczęściej stosowana składnia polecenia `kill` wygląda następująco:

```
kill [-signal] PID...
```

Jeśli nie sprecyzujemy żadnego sygnału w wierszu poleceń, to domyślnie wysłany zostanie sygnał `TERM` (*Terminate*). Polecenie `kill` jest najczęściej wykorzystywane do wysyłania sygnałów opisanych w tabeli 10.4.

Tabela 10.4. Często używane sygnały

Numer	Nazwa	Znaczenie
1	HUP	<i>Hung up.</i> Jest to pozostałość ze starych dobrych czasów, gdy terminale były podłączone do komputerów zdalnych poprzez kable telefoniczne i modemy. Sygnał jest wykorzystywany do wskazania programów, które terminal kontrolny „zawiesił”. Efekt działania tego sygnału można zaobserwować poprzez zamknięcie sesji terminala. Program pierwszoplanowy działający w terminalu otrzyma sygnał i zakończy działanie. Sygnał ten jest też wykorzystywany przez wiele demonów do ponownej inicjalizacji. Gdy sygnał ten zostanie wysłany do demona, demon będzie uruchomiony ponownie i ponownie odczyta swój plik konfiguracyjny. Przykładem demona, który w ten sposób korzysta z sygnału HUP, jest serwer Apache.
2	INT	<i>Interrupt.</i> Wywołuje takie samo działanie jak kombinacja klawiszy <i>Ctrl+C</i> użyta w terminalu. Zwykle zatrzymuje działanie programu.
9	KILL	<i>Kill.</i> Jest to sygnał specjalny. Zwykle programy mogą obsługiwać wysyłane do nich sygnały na różne sposoby, a także je ignorować. Natomiast sygnał KILL nie jest tak naprawdę wysyłany do programu docelowego. W tym przypadku to jądro natychmiast kończy proces. Gdy proces zostanie w ten sposób zatrzymany, nie ma żadnej możliwości „posprzątania” po sobie lub zapisania efektów działania. Z tego powodu sygnału KILL należy używać tylko jako ostatniej deski ratunku, gdy inne sygnały służące do kończenia procesu zawiodą.
15	TERM	<i>Terminate.</i> Jest to domyślny sygnał wysyłany przez polecenie <i>kill</i> . Jeśli program jest nadal wystarczająco „żywy”, aby przyjmować sygnały, zakończy swoje działanie.
18	CONT	<i>Continue.</i> Sygnał ten wznawia proces zatrzymany przez sygnał STOP.
19	STOP	<i>Stop.</i> Ten sygnał sprawia, że proces wstrzymuje działanie, lecz nie kończy pracy. Podobnie jak KILL, również ten sygnał nie jest przesyłany do procesu docelowego i dlatego nie może być zignorowany.

Wypróbujmy teraz działanie polecenia `kill`:

```
[me@linuxbox ~]$ xlogo &
[1] 13546
[me@linuxbox ~]$ kill -1 13546
[1]+  Hangup                  xlogo
```

W powyższym przykładzie uruchamiamy w tle program `xlogo`, a następnie wysyłamy do niego sygnał HUP poprzez polecenie `kill`. Program `xlogo` kończy działanie, a powłoka informuje, że proces w tle otrzymał sygnał *Hangup*. Możliwe, że zanim zobaczymy wiadomość, trzeba będzie kilkakrotnie nacisnąć klawisz *Enter*. Zauważmy, że sygnały można określić zarówno przez liczbę, jak i przez nazwę, włącznie z nazwą o przedrostku *SIG*:

```
[me@linuxbox ~]$ xlogo &
[1] 13601
[me@linuxbox ~]$ kill -INT 13601
```

```
[1]+ Interrupt                xlogo
[me@linuxbox ~]$ xlogo &
[1] 13608
[me@linuxbox ~]$ kill -SIGINT 13608
[1]+ Interrupt                xlogo
```

Powtórzmy powyższy przykład i wypróbujmy inne sygnały. Pamiętajmy, że zamiast numerów PID możemy skorzystać z numerów zadań.

Procesy, podobnie jak pliki, posiadają właścicieli. Jeśli jesteśmy właścicielem procesu (lub jeśli jesteśmy użytkownikiem uprzywilejowanym), możemy wysłać do niego sygnały z wykorzystaniem polecenia `kill`.

Oprócz sygnałów wymienionych w tabeli 10.4, które są najczęściej wykorzystywane wraz z poleceniem `kill`, system często korzysta z innych sygnałów. W tabeli 10.5 opisano inne często używane sygnały.

Tabela 10.5. Inne często wykorzystywane sygnały

Numer	Nazwa	Znaczenie
3	QUIT	Wyjście.
11	SEGV	Naruszenie ochrony pamięci. Sygnał ten jest wysyłany, jeśli program korzysta z pamięci w nieuprawniony sposób, czyli gdy próbuje zapisać dane w miejscu, do którego nie jest uprawniony.
20	TSTP	Zatrzymanie terminala. Sygnał ten jest wysyłany przez terminal, gdy została użyta kombinacja <code>Ctrl+Z</code> . Inaczej niż w przypadku sygnału STOP, sygnał TSTP jest przechwytywany przez program, lecz może zostać zignorowany.
28	WINCH	Zmiana okna. Jest to sygnał wysyłany przez system podczas zmiany rozmiaru okna. Niektóre programy, takie jak <code>top</code> i <code>less</code> , będą odpowiadać na ten sygnał poprzez ponowne wyświetlenie wyników, aby dostosować się do nowych wymiarów okna.

Informacja dla ciekawskich — pełną listę sygnałów można wyświetlić, wykonując następujące polecenie:

```
[me@linuxbox ~]$ kill -l
```

Wysyłanie sygnałów do wielu procesów za pomocą polecenia `killall`

Możemy także wysłać sygnały do wielu procesów, dotyczących określonego programu lub należących do określonego użytkownika. W tym celu stosujemy polecenie `killall`, którego składnia wygląda następująco:

```
killall [-u użytkownik] [-sygnał] nazwa...
```

Zademonstrujemy działanie tego polecenia. W tym celu uruchomimy kilka instancji programu `xlogo` i zakończymy ich działanie:

```
[me@linuxbox ~]$ xlogo &
[1] 18801
[me@linuxbox ~]$ xlogo &
[2] 18802
[me@linuxbox ~]$ killall xlogo
[1]- Terminated          xlogo
[2]+ Terminated          xlogo
```

Pamiętajmy, że aby wysyłać sygnały do procesów, które do nas nie należą, musimy posiadać uprawnienia użytkownika uprzywilejowanego, podobnie jak w przypadku polecenia `kill`.

Więcej poleceń dotyczących procesów

Ponieważ monitorowanie procesów jest ważnym zadaniem administracyjnym, istnieje wiele poleceń do tego przeznaczonych. W tabeli 10.6 przedstawiono kilka poleceń wartych wypróbowania.

Tabela 10.6. Inne polecenia dotyczące procesów

Polecenie	Opis
<code>pstree</code>	Wyświetla procesy w postaci drzewa, pokazując zależności między procesami macierzystymi a potomnymi.
<code>vmstat</code>	Wyświetla informacje o wykorzystaniu zasobów systemu, włącznie z pamięcią, przestrzenią wymiany oraz dysku I/O. Aby wyświetlić stale uaktualnianą listę, należy podać po poleceniu czas opóźnienia (w sekundach) dla uaktualnienia (np. <code>vmstat 5</code>). Działanie programu kończymy, używając kombinacji <code>Ctrl+C</code> .
<code>xload</code>	Program graficzny rysujący wykres obciążenia systemu w czasie.
<code>tload</code>	Podobny do programu <code>xload</code> , jednak rysuje wykres w terminalu. Działanie programu kończymy, używając kombinacji <code>Ctrl+C</code> .

Skorowidz

A

adres
 IP, 210, 213
 URI, 210, 214
algorytm kompresji, *Patrz:* kompresja algorytm
alias, 66, 72, 73, 142, 143, 385
American National Standards Institute, *Patrz:* ANSI
ANSI, 174, 175
ASCII, 43, 263
assembler, 344, 345
atak
 man-in-the-middle, 216
 temp race attack, 495
AWK, 475

B

balkanizacja, 266
basic regular expressions, *Patrz:* wyrażenie
 regularne podstawowe
 program, 475
biblioteka, 345
 mktemp, 495
 Readline, 102, 103
 współdzielona, 46, 47, 183

bit

 bucket, 79
 setgid, 118
 setuid, 118
 zaczepienia, 119
błąd, 77
 logiczny, 427
 składniowy, 423, 424, 425
Bourne Steve, 29
BRE, *Patrz:* wyrażenie regularne podstawowe
bufor
 drukowania, 197
 FIFO, 498
 kill-ring, 103
buforowanie, 197

C

CentOS, 182, 184, 339
CLI, *Patrz:* interfejs wiersza poleceń
COBOL, 344
Common Unix Printing System, *Patrz:* CUPS
coreutils, 289
CPU, 344
CUPS, 334, 335, 339, 342
cytowanie, 96

czas, 173
czcionka, 332, 333

D

dane
baza relacyjna, 291
tabelaryczne, 281
weryfikacja, 410
współdzielone, 47
data, 31, 173
Debian, 182, 184
repozytorium, 343
debugowanie, 431
demon, 128
DHCP, 213
Digital Rights Management, *Patrz:* DRM
dokument
HTML, 365
włączony, 372, 373, 374, 409
dokumentacja, 72
Dolphin, 52
dopisek, 93
dowiązanie
miękkie, *Patrz:* dowiązanie symboliczne
przerwane, 62
symboliczne, 47, 56, 63
tworzenie, 55, 60, 61, 63
twarde, 48, 56, 60
tworzenie, 55, 59
DRM, 183
drukarka
fizyczna, 341
głowicowa, 333
graficzna, 333
laserowa, 333
PostScript, 334, 337
rozetkowa, 332
wirtualna, 341
drukowanie, 331, 332, 333, 334, 335, 339
Berkeley, 335, 336, 342
do pliku, 337
kolejka, 334, 339
status, 342
kończenie zadań, 342
LPD, *Patrz:* drukowanie Berkeley
SysV, 335, 337, 342
dysk
CD
dźwiękowy, 206
obraz, 205, 206, 207
CD-RW, 207

ilość wolnego miejsca, 32
RAM obraz wstępny, 46
dyskietka, 204
dystrybucja, 182, 183
Fedora, *Patrz:* Fedora
OpenSUSE, *Patrz:* OpenSUSE
Ubuntu, *Patrz:* Ubuntu

E

edytor
tekstu, 147, 148, 149, 153
graficzny, 147, 148
tekstowy, 147, 148
vi, *Patrz:* vi
wierszowy, 154
wizualny, 154
ekran, 144
emacs, 147, 185
e-mail, 276
emulator terminala, 29, 30, 31, 144
ERE, *Patrz:* wyrażenie regularne rozszerzone
extended regular expressions, *Patrz:* wyrażenie
regularne rozszerzone

F

Fedora, 24, 111, 182, 184, 339
filtr, 82
Foresight, 182
FORTRAN, 344, 458
FTP, 214
funkcja
liczbowa, 394, 395
opakowująca, 446
plikowa, 390, 391
powłoki, *Patrz:* powłoka funkcja
tekstowa, 393
operator, 395
zastępcza, 382

G

gedit, 147, 148
Gentoo, 182
Ghostscript, 334
gid, *Patrz:* użytkownik identyfikator grupy
głównej
globbing, *Patrz:* wieloznacznik
GNOME
dowiązanie symboliczne, 63
edytor tekstu, 147
menedżer plików, *Patrz:* Nautilus

godzina, 31
GUI, 22, 102

H

historia poleceń, 30, 105, 107
interpretacja, 106, 107, 108
przeszukiwanie, 106
host
nazwa, 173, 210
zdalny, 216, 219
localhost, 217

I

IEEE, 266
IFS, *Patrz:* zmienna IFS
Institute of Electrical and Electronics Engineers,
Patrz: IEEE
interfejs
Ethernetu, 213
pętli zwrotnej, 213
sieciowy, 212
użytkownika graficzny, *Patrz:* GUI
wiersza poleceń, 22, 357
wirtualny, 213
Internal Field Separator, *Patrz:* zmienna IFS
interpreter, 345, 357

J

jądro, 128, 189
język
assemblera, 344
C, 345, 458
C++, 345
COBOL, *Patrz:* COBOL
FORTRAN, *Patrz:* FORTRAN
interpretowany, 345
maszynowy, 344
opisu strony, 334
Perl, *Patrz:* Perl
PHP, *Patrz:* PHP
PostScript, *Patrz:* PostScript
Python, *Patrz:* Python
Ruby, *Patrz:* Ruby
skryptowy, 345, 357
wysokiego poziomu, 344
znaczników, 276, 325
HTML, 276
XML, 276
Joy Bill, 154

K

kalendarz, 31
katalog, 45
/, 46
/bin, 46
/boot, 46
/dev, 46
/etc, 46, 128, 147
/home, 46
/home/me/bin, 360
/lib, 46, 345
/lost+found, 46
/media, 46
/mnt, 46
/opt, 46
/proc, 46
/root, 46
/sbin, 46
/tmp, 47, 495
/usr, 47
/usr/bin, 35, 47
/usr/lib, 47, 345
/usr/local, 47
/usr/local/bin, 361
/usr/local/src, 347
/usr/sbin, 47
/usr/share, 47
/usr/share/dict, 260
/usr/share/doc, 47
/usr/share/man, 326
/usr/src, 347
/var, 47
/var/log, 47
~/bin, 361
domowy, 34, 46, 144
główny, 33, 46
kopiowanie, 52
nadrzędny, 34, 36
nazwa ukryta, 37
przenoszenie, 59
roboczy
bieżący, 34, 35, 144, 173
wcześniejszy, 144
struktura hierarchiczna, 33
katalog
tworzenie, 52, 57
uprawnienia, *Patrz:* uprawnienia katalogu
usuwanie, 54, 61
właściciel, 110
zmiana, 122
współdzielony, 124
wyszukiwanie, 226

kate, 147
KDE
 dowiązanie symboliczne, 63
 edytor tekstu, 147
 menedżer plików, *Patrz:* Dolphin, Konqueror
kedit, 147
killing, 103
klawiatura, 403, 410, 412
klient SSH, 216, 217, 221
Knuth Donald, 324
kod
 binarny, 344
 źródłowy własny, 347
 dystrybucji, 347
 wielu użytkowników, 347
kodowanie długości serii, 240
komentarz, 145, 149, 151, 431
kompilacja, 343, 344, 346
kompilator, 345
 C, *Patrz:* kompilator gcc
 gcc, 346, 350
 kompresja, 239, 240, 247, 248
komunikat o błędzie, 77, 110
Konqueror, 52, 116
konsola wirtualna, 32, 102
kotwica, 259
kursora przemieszczanie, 102, 176, 177
kwrite, 147

L, ł

licencja GPL, 25
linkowanie, 345
Linspire, 182
Linux
 dystrybucja, *Patrz:* dystrybucja
 jądro, 46
 uruchamianie, 25
lista plików, 39, 40
 długi format, 40, 41, 42
literał, 258, 264
Lukyanov Alexander, 215
LVM, 191
łańcuch
 formatowania, 321
 tekstowy, *Patrz:* tekst łańcuch
 włączony, 409

M

Mandriva, 182
menedżer plików
 GNOME, *Patrz:* Nautilus
 KDE, *Patrz:* Dolphin, Konqueror

menu, 411
metaklawisz, 103
 uzupełnianie, 105
metasekwencja, 258
metaznak, 258, 260, 264, 266
montowanie, 192, 194
 systemu plików, 193
mysz, 31

N

nano, 147, 148, 149, 153
narzędzie systemowe bit bucket, *Patrz:* bit bucket
Nautilus, 51, 116
notacja ósemkowa, 113, 114, 115, 467

O

obraz dysku CD, 205
odmontowywanie, 196, 197
okna aktywowanie, 31
OpenSSH, 217, 220
OpenSSH-server, 217
OpenSUSE, 24, 182
operator
 %, 92
 &&, 400
 *, 92
 **, 92
 /, 92
 |, 82
 ||, 400
 +, 92
 ++, 471
 <<<, 409
 =~ , 395
 ==, 396, 397
 arytmetyczny, 92, 468
 bitowy, 472
 funkcji tekstowej, 395
 jednoargumentowy, 468
 +, 468
 logiczny, 227, 229, 231, 232, 397, 473
 modulo, 468, 469
 porównania, 472, 473
 potrójny, 473
 przekierowania
 <, 81, 420
 >, 76, 77
 >>, 77
 przypisania, 470
 sterowania, 400
oprogramowanie, 122

P

- page-description language, *Patrz:* PDL
- pakiet
 - cdrttools, 206
 - coreutils, 289
 - gcc, 346
 - instalowanie, 185
 - lista, 187
 - make, 346
 - makr, 325, 326
 - metadane, 182, 183
 - nazwa, 184
 - opis, 188
 - plik, *Patrz:* plik pakietu
 - texlive, 325
 - uaktualnianie, 186
 - usuwanie, 186
 - wyszukiwanie, 185
 - zarządzanie, 181, 182, 183, 184
 - narzędzia, 184, 185, 186, 188
- pamięć operacyjna, 32
- parametr
 - interpretacja, 94, 96, 97, 461, 462, 464, 466
 - pozycyjny, 441, 444, 445, 464
 - , 446, 447
 - @, 446, 447
 - zarządzanie, 446
- parent directory, *Patrz:* katalog nadrzędny
- partycja, 46, 192, 193, 199
 - tworzenie, 200
 - usuwanie, 200
- pasek adresu wieloznacznik, *Patrz:* wieloznacznik
- PCLinuxOS, 182
- PDL, 334
- Perl, 345, 475
- pętla, 415
 - for, 455, 457, 458
 - until, 420
 - while, 416, 417, 419, 420, 425
- PHP, 345
- PID, *Patrz:* proces identyfikator
- plik
 - .bash_history, 105
 - .bash_profile, 147
 - .bashrc, 147, 149, 150, 177, 252
 - .bz2, 242, 243
 - .c, 348
 - .gz, 242
 - .h, 348
 - .profile, 147
 - .tar, 243
 - .tar.gz, 247
 - .tbz, 247
 - .tgz, 243, 247
 - .zip, 248, 249
 - /boot/grub/grub.conf, 46
 - /boot/vmlinuz, 46
 - /dev/null, 79
 - /etc/bash.bashrc, 145
 - /etc/crontab, 46
 - /etc/fstab, 46, 192, 193, 203
 - /etc/group, 111
 - /etc/passwd, 46, 111
 - /etc/profile, 145
 - /etc/shadow, 111
 - /etc/sudoers, 120
 - /var/log/messages, 198
 - ~/bash_login, 145
 - ~/bash_profile, 145
 - ~/bashrc, 145
 - ~/profile, 145
 - ANSI.SYS, 175
 - archiwizowanie, 239, 243, 244, 247, 248
 - atrybut, 39
 - binarny, 46
 - deskryptora, 78
 - diction.o, 351
 - dokumentacji, 47
 - dźwiękowy, 47
 - getopt.h, 349
 - getopt.o, 351
 - install, 353
 - ISO, 205, 206, 207
 - JPEG, 240
 - kompresja, *Patrz:* kompresja
 - konfiguracyjny, 43, 46, 47, 148, 151, 350
 - edytowanie, 122
 - konwersja formatu, 328
 - kopiowanie, 52, 57, 58
 - liczba słów, 83
 - lista, *Patrz:* lista plików
 - loga, 47
 - makefile, 350, 351
 - menu.lst, 46
 - misc.o, 351
 - MP3, 240
 - nagłówkowy, 348
 - nazwa, 234
 - ukryta, 37
 - zmiana, 53, 58
 - niewymienialny, 119
 - pakietu, 182, 183, 184, 185, 186
 - przenoszenie, 53, 58

plik

- README, 72
 - sentence.o, 351
 - startowy, 143, 144, 145, 151
 - użytkownika, 145
 - synchronizacja, 239, 251
 - tar, 347, 348
 - tarball, 347
 - tekstowy, 43, 44
 - porównywanie, 293, 294, 296
 - zawartość, 43
 - tryb
 - maska bitów, 116, 118
 - tymczasowy, 47, 493, 495
 - nazwa, 495
 - typ, 37, 42, 112
 - ukryty, 91
 - uprawnienia, *Patrz*: uprawnienia pliku
 - usuwanie, 54, 61
 - właściciel, 110
 - zmiana, 122
 - wykonywalny
 - lokalizacja, 66
 - wyszukiwanie, 224, 225, 226, 235
- system, 45
- podpowłoka, 489, 490
- podręcznik programu, 68, 69, 70
 - przeszukiwanie, 69
- podstawianie wyników procesów, 491
- polecenie, 65
- a2ps, 331, 337, 338, 339, 340
 - alias, 72, 143
 - apropos, 69
 - argument, 40
 - aspell, 276, 307
 - awk, 307
 - bg, 134
 - break, 418
 - bzip2, 239, 242
 - bzip2recover, 243
 - cal, 31
 - cancel, 331, 342
 - case, *Patrz*: polecenie złożone case
 - cat, 80, 82, 275, 277, 278, 314
 - cd, 35, 91
 - skrót, 37
 - cdrdao, 206
 - cdrecord, 192, 206
 - chgrp, 123
 - chmod, 113, 115, 116, 119
 - chown, 122, 123
 - clear, 101, 102
 - comm, 275, 293
 - configure, 349, 350
 - continue, 418
 - cp, 50, 52, 53, 57, 58, 220
 - cut, 275, 287, 289, 290
 - date, 31
 - dd, 192, 204, 205
 - declare, 371, 480
 - df, 32
 - dformat, 204
 - diction, 347
 - diff, 275, 294
 - format kontekstowy, 296
 - format zuniifikowany, 297
 - dokumentacja, 67
 - dos2unix, 278
 - echo, 89, 91, 266, 372, 434
 - egrep, 267
 - egrep.m, 265
 - enscript, 339
 - ex, 157, 164
 - exit, 32
 - expand, 289
 - extract, 289
 - fdformat, 192
 - fdisk, 191, 200
 - fg, 134
 - file, 42
 - find, 223, 225, 226, 227, 230, 234, 235, 246, 247, 271
 - akcja, 230, 232
 - logika, 230
 - opcje, 237
 - operator logiczny, 229
 - testy, 228
 - typ pliku, 226
 - fmt, 313, 317, 319
 - fold, 313, 317
 - for, *Patrz*: pętla for
 - free, 32, 197
 - fsck, 192, 203, 204
 - ftp, 209, 214
 - interaktywne, 215
 - genisoimage, 192, 206
 - grep, 83, 256, 258, 264, 266
 - opcje, 257
 - wersja, 265
 - groff, 313
 - grupowanie, 489
 - grupowe, 489, 490, 491
 - gunzip, 240, 242
 - gzip, 239, 240, 241

head, 84
 help, 67
 history, 101
 id, 110
 info, 70, 71
 interpretacja, 89
 ispell, 307
 join, 275, 291, 292
 kill, 135, 136
 killall, 137
 kropki, 360
 less, 43, 44, 77, 82, 272, 276, *Patrz też:*
 program less
 lftp, 209, 215
 ln, 55
 locale, 265
 locate, 223, 224, 225, 272
 lp, 331, 337, 338
 lpq, 331, 342
 lpr, 331, 336, 337
 lprm, 331, 342
 lpstat, 331, 341
 ls, 35, 39, 55, 59, 73, 78, 235
 -l, 40, 41
 opcja, 41
 --reverse, 41
 make, 344, 346, 350, 351, 352, 353
 maksymalny rozmiar, 234
 man, 68, 69, 70, 103, 326
 md5sum, 192, 208
 mkdir, 52, 57, 235
 mkfifo, 498
 mkfs, 192, 202
 mkisofs, 192, 206
 mount, 191, 193
 mv, 53, 54, 58
 netstat, 209, 212
 nl, 313, 314, 315
 opcja, 40, 41
 passwd, 126
 paste, 275, 290, 291
 patch, 275, 296
 perl, 307
 ping, 209, 210
 pr, 313, 320, 331, 335, 336
 printenv, 142, 143
 printf, 313, 321, 322, 323
 ps, 128, 129, 130
 pstree, 138
 read, 404, 406, 407
 opcje, 405, 407
 potok, 409
 rlogin, 216
 rm, 54, 55, 61
 rsync, 239, 251, 252
 zdalnie, 253
 scp, 210, 220
 sed, 276, 300, 301, 303, 305, 307
 set, 142, 143
 sftp, 210, 220
 shift, 443, 444
 sort, 82, 83, 275, 279, 285, 290
 opcje, 280
 source, 360
 ssh, 209, 217
 stat, 235
 su, 120, 121, 122
 sudo, 120, 121, 122
 uprawnienia, 122
 tail, 84, 85
 tar, 239, 243, 244, 245, 246, 247, 249, 347
 tee, 85
 telnet, 216
 test, 390, 391, 393, 394, 395, 397, 399, 470
 tload, 138
 top, 130, 132
 touch, 223, 235
 tr, 275, 298, 299
 tracepath, 211
 traceroute, 209, 211
 trap, 494
 tworzenie, 72
 typ, 66
 type, 66
 umask, 116, 125
 umount, 191
 uniq, 83, 275, 285, 286
 unix2dos, 278
 unset, 485
 until, *Patrz:* pętla until
 unzip, 249
 updatedb, 225
 vim, 272
 vmstat, 138
 wait, 496
 wbudowane, 404
 wbudowane powłoki, 66
 wc, 83
 wget, 209, 216
 whatis, 70
 which, 66, 67
 which cp, 95
 while, *Patrz:* pętla while
 wodim, 192, 206, 207

- połączenie
 - wyszukiwania pakietów, 185
 - xargs, 223, 233, 234
 - xload, 138
 - zip, 239, 248, 249, 250
 - złożone, 425
 - (()), 396, 397, 467, 470, 472
 - [[]], 395, 399, 470
 - case, 435, 437
 - if, 146, 387, 388, 389, 390
 - pętla, *Patrz:* pętla wielokrotnego wyboru, 435
- Portable Operating System Interface, *Patrz:* POSIX
- POSIX, 263, 264, 266, 399
- PostScript, 276, 334
 - interpreter, *Patrz:* Ghostscript
- potok, 82, 409
 - filtr, 335
 - nazwany, 497, 498
 - rozdzielenie, 85
- powłoka, 357
 - bash, 29, 66, 67, 141, 400, 489
 - funkcja, 66, 82, 142, 378, 379, 380, 385
 - sterowaniem zadaniami, *Patrz:* zadanie sterowanie
 - logowania, 144
 - połączenie wbudowane, *Patrz:* połączenie wbudowane powłoki
 - sesja, *Patrz:* sesja powłoki
 - sh, 29
 - skrypt, *Patrz:* skrypt powłoki
 - udostępnianie, 29
 - wersja, 173
 - wydanie, 173
 - zmienna, *Patrz:* zmienna powłoki
- prawa dostępu, *Patrz:* uprawnienia
- prawami cyfrowymi zarządzanie, *Patrz:* DRM
- preambuła, 93
- primary group ID, *Patrz:* użytkownik
 - identyfikator grupy głównej
- proces, 127, 138
 - identyfikator, 128, 133, 135
 - macierzysty, 128, 138
 - potomny, 128, 138
 - przywracanie do pierwszego planu, 134
 - stan, 129
 - sterowanie, 131
 - umieszczanie w tle, 133
 - wstrzymywanie, 134
 - wyświetlanie, 128
 - zakończenie, 135
 - zatrzymywanie, 132
- procesor
 - komputera, *Patrz:* CPU
 - tekstu, 147, 325
- process id, *Patrz:* proces identyfikator
- program, *Patrz też:* polecenie
 - apt-cache, 185, 188
 - apt-get, 184, 185, 186
 - aptitude, 184
 - bash, 143
 - dpkg, 184, 185, 187, 188
 - gedit, 132
 - groff, 325, 326, 328
 - init, 128
 - kompilowanie, *Patrz:* kompilacja
 - kwirte, 132
 - less, 44, 45, *Patrz też:* polecenie less
 - linkowanie, *Patrz:* linkowanie
 - ls, 77
 - mktemp, 495
 - more, 45
 - nroff, 324, 325
 - PuTTY, 221
 - roff, 324, 325
 - rozruchowy
 - konfiguracja, 46
 - rpm, 184, 185, 187, 188
 - script, 108
 - troff, 324, 325
 - wstępnie skompilowany, 343
 - wykonywalny, 65
 - xlogo, 131, 132
 - yum, 184, 185, 186, 188
- programowanie defensywne, 427, 431
- projektowanie zstępujące, 377
- protokół
 - DHCP, 213
 - FTP, 214
 - HTTP, 215
 - SFTP, 221
 - SSH, 216, 219
- przekierowanie, 76, 491
 - standardowego strumienia
 - błędów, 78, 79
 - wejścia, 80, 81
 - wyjścia, 76, 78, 79, 82
- przenośność, 399
- przypadek testowy, 430
- przypisanie wartości, 469
- pułapka, 493
- punkt montowania, 194
 - nośników wymiennych, 46
 - tworzenie, 195

PuTTY, 221
Python, 345

R

RAID, 191
Red Hat, 182, 184, 217
Red Hat Enterprise Linux, 182, 184
rekord, 281
relacyjna baza danych, 291
repozytorium
 deweloperskie, 183
 testowe, 183
root directory, *Patrz:* katalog główny
ROT13, 299
rozgałęzienie, 387, 400
Ruby, 345

S, Ś

Schilling Jörg, 206
serwer
 anonimowy FTP, 214
 SSH, 216, 217
sesja
 powłoki
 logowania, 144, 145
 niesłużącej do logowania, 144, 145
 uprawnienia, 30
 zamykanie, 32
Seward Julian, 242
sieć VPN, 219
skrypt, 43, 66, 128, 142, 151
 inicjalizacyjny, 128
 konfiguracyjny, 145
 poinstalacyjny, 183
 potomny, 497
 powłoki, 46, 357
 format, 358
 lokalizacja, 359, 361
 tworzenie, 357, 358
 uprawnienia do wykonania, 359
 przedinstalacyjny, 183
 uprawnienia do wykonania, 359
 włączony, *Patrz:* dokument włączony
Slackware, 182
słownik, 260
SSH, 216, 219
stacja dyskiety, 204
Stallman Richard, 21, 25, 148, 266, 346
stała, 370
status wyjścia, 388, 389, 417
stderr, 76
stdin, 76

stdout, 76
sterownik urządzenia, 189
strona
 info, 70
 logiczna, 314
 man, *Patrz:* podręcznik programu
strumień standardowy
 błędów, 76, 77
 wejścia, 76, 80, 279, 404, 420, 491
 wyjścia, 76, 77, 80, 85, 491
sygnał, 135, 493
 CONT, 136
 HUP, 136
 INT, 136
 KILL, 136
 lista, 137
 obsługa, 494, 495
 QUIT, 137
 SEGV, 137
 STOP, 136
 TERM, 136
sygnał, 135, 493
 TSTP, 135, 137
 WINCH, 137
sygnał dźwiękowy, 104, 173
symlink, *Patrz:* dowiązanie symboliczne
system
 jądro, *Patrz:* jądro
 obciążenie, 138
 operacyjny
 GNU/Linux, 25
 ósemkowy, 113, 114, 115, 467
 plików, 33
 drzewo, 34
 FAT32, 199
 hierarchia, 44
 tworzenie, 199
 uszkodzenie, 46
 wirtualny, 46
 sieciowy, 209
 składu TEX, *Patrz:* TEX
 szesnastkowy, 114, 115, 467
 wbudowany, 344
 wieloużytkownikowy, 109, 122, 496
 wielozadaniowy, 496
 wykorzystanie zasobów, 138
 X Window, 31, 220
zarządzania
 drukami, 334
 obsługą interfejsu graficznego,
 Patrz: system X Window
 pakietami, *Patrz:* pakiet zarządzanie

szyfr ROT13, 299
ścieżka, 35
 bezwzględna, 35
 interpretacja, 90, 91, 96
 względna, 36, 37
środowisko, 141
 GNOME, 29, 31
 KDE, 29, 31
 konfiguracja, 142
 modyfikowanie, 147

T

tablica
 dwuwymiarowa, 479
 element, 479, 481, 483
 dodawanie, 485
 liczba, 484
 przypisanie wartości, 480
 sortowanie, 485
 trasowania sieci jądra, 213
 usuwanie, 485
Tatham Simon, 221
tekst, 276, 313
 dodawanie kolumny, 290
 dzielenie na strony, 320, 321
 edycja, 102, 103
 interaktywna, 307
 nieinteraktywna, 298, 299, 300, 305, 307
 format ASCII, *Patrz:* ASCII
 formatowanie, 317, 321, 322, 324, 325
 kodowanie metodą ROT13, 299
 łańcuch, 73, 93, 224, 358, 367, 371, 392, 433,
 457, 461
 długość, 464
 pusty, 96, 427
 weryfikacja, 392
 włączony, 409
 wyodrębnianie, 465
 łączenie, 291, 292
 maskowanie, 299
 MS-DOS, 278
 porównywanie, 293, 294, 296
 sortowanie, 279, 280, 281, 282, 283, 285
 transliteracja, 298
 uniksowy, 278
 wiersz
 numerowanie, 314
 zawijanie, 317
 wyodrębnianie, 287, 288, 289
terminal, 103, 144, 173, 175, 498
 dalekopisowy, 154
 emulator, *Patrz:* emulator terminala

 wideo, 154, 175
 wirtualny, 32
Terminal Stop, *Patrz:* sygnał TSTP
emulator, 175
test prawdy arytmetycznej, 396
testowanie, 429
 element zastępczy, 429
 przypadek testowy, *Patrz:* przypadek testowy
 śledzenie, *Patrz:* śledzenie
 wyświetlanie wartości zmiennej, 434
TEX, 324, 325
Text Editor, 147
token, 373
Torvalds Linus, 21
tunelowanie, 219, 220

U

Ubuntu, 24, 41, 111, 122, 182, 184, 217, 339, 361
uid, *Patrz:* użytkownik identyfikator
Unix
 standaryzacja, 266
 termcap, 175
 terminfo, 175
uprawnienia, 110, 111, 115
 atrybut, 112, 113
 do odczytu, 111
 do wykonywania, 111
 do zapisu, 111
 katalogu, 113
 pliku
 domyślne, 116, 118
 zmiana, 113
 specjalne, 118
 zmiana, 113
urządzenie
 klonowanie, 204
 montowanie, *Patrz:* montowanie, punkt
 montowania
 nazwa, 196
 pamięciowe, 191, 192, 198, 240
 wirtualne, 191
 sieciowe, 191
 węzeł, *Patrz:* węzeł urządzeń
user ID, *Patrz:* użytkownik identyfikator
usługa systemowa, 46, 128
ustawienia językowe, 265
uzupełnianie, 104, 105
 programowalne, 105
użytkownik
 grupa, 110
 hasło, 126

- identyfikator, 111
 - egid, 118
 - euclid, 118
 - grupy głównej, 111
- katalog domowy, 46
- nazwa, 173
- plik startowy, 145
- uprzywilejowany, 25, 111, 118, 120, 122, 173, 495
- uprawnienia, 30
- zmiana tożsamości, 119, 120, 121

V

- vi, 147, 153
 - edycja wielu plików, 166, 167, 168, 169
 - historia, 154
 - kursor zmiana położenia, 158
 - łączenie wierszy, 163
 - plik
 - tworzenie, 155
 - zapisywanie, 169
 - tekst
 - dodawanie, 159
 - kopiowanie, 162, 168, 169
 - usuwanie, 160
 - wklejanie, 162, 168, 169
 - tryb, 156
 - edycji, 156
 - uruchamianie, 154
 - wyszukiwanie, 163, 164, 165
 - zatrzymywanie, 154
- vim, 148, 157
 - konfigurowanie na potrzeby skryptów, 362
- VPN, 219

W

- węzeł urządzeń, 46
- wiadomość e-mail, 276
- wieloznacznik, 50, 51, 52, 55, 90, 465
 - kolejność, 80
- wiersz
 - polecień, 22, 23
 - edycja, 102
 - liczba argumentów, 442, 443, 444
 - parametr pozycyjny, *Patrz:* parametr pozycyjny
- world, 110
- wyrażenie
 - arytmetyczne, 92
 - interpretacja, 92, 96, 97, 467

- przypisanie wartości, 469
- zagnieżdżone, 93
- nawiasowe, 260, 266
- regularne, 255, 270, 271, 396
 - alternatywa, 266, 267
 - kotwica, 259
 - kwantyfikator, 267, 268, 269
 - metaznak, *Patrz:* metaznak
 - podstawowe, 264
 - rozszerzone, 264, 266, 267, 268, 269
 - rozszerzone regex, 395

WYSIWYG, 332

- wzorzec
 - tekstowy, 83
 - uprawnień pożądaných, 113

X

- X Window, 31, 220
- Xandros, 182

Y

- yanking, 103

Z

- zadanie
 - liczba, 173
 - sterowanie, 133
- zależność, 184
- zmienna, 369
 - \$#, 442
 - globalna, 381
 - IFS, 408
 - lokalna, 381, 382
 - nazwa, 370, 464
 - nieistniejąca, 462
 - PATH, 146, 147, 360, 379
 - powłoki, 141
 - przypisywanie wartości, 371
 - pusta, 369, 462
 - środowiskowa, 141, 142, 144, 147
 - DISPLAY, 144
 - EDITOR, 144
 - HOME, 144
 - LANG, 144, 265
 - PAGER, 144
 - PATH, 144
 - PS1, 144, 172
 - PWD, 144
 - SHELL, 144
 - TERM, 144

zmienna
 TZ, 144
 USER, 144
 tablicowa, 484
 nazwa, 480
 tworzenie, 369

znak
 ', 98
 !, 107, 108
 !!, 108
 !?, 108
 ", 96, 99, 100, 258, 265
 "', 96, 97
 #, 30, 145, 151, 173
 \$, 30, 96, 172, 173, 258, 259, 264
 %, 92
 &, 133
 &&, 400
 (), 258, 264, 265, 267, 399
 *, 90, 91, 92, 258, 264, 268, 444, 464, 483
 **, 92
 ., 36, 37, 258, 264, 360
 ..., 36
 /, 92
 ?, 258, 264, 268
 @, 172, 464, 465, 483
 [], 67, 258, 260, 264
 ^, 258, 259, 260, 261, 264
 `, 96
 {}, 96, 258, 264, 265, 269, 371
 interpretacja, 93
 |, 82, 258, 264, 266
 ||, 400
 ~, 96, 156, *Patrz:* znak tyldy
 +, 92, 258, 264, 269, 468
 <, 81, 399
 <<<, 409
 =, 470
 =~ , 395
 ==, 396, 397, 470
 >, 76, 77, 367, 399
 >>, 77
 ASCII, 43, 263
 Backspace, 99
 biały, 93, 96
 cudzysłowu podwójnego, *Patrz:* znak ""
 cudzysłowu, *Patrz:* znak ''
 interpretowanie, 99
 klasa, 50
 literal, *Patrz:* literal
 niedrukowalny, 173
 nowego wiersza, 97, 173, 278
 null, 234
 POSIX, 262, 263, 264
 powrotu karetki, 173, 278
 separatora, 97, 98, 234, 282
 spacji, 92, 97, 173, 234
 specjalny ANSI, 174
 sterujący, 99, 277
 średnika, 425
 tabulatora, 97, 99, 277, 289
 transliteracja, 298
 tyldy, 91, 96
 zachęty, 30, 144, 171, 178, 367
 \!, 173
 \#, 173
 \\$, 173
 \@, 173
 \[, 173
 \], 173
 \a, 173
 \A, 173
 atrybut, 176
 \d, 173
 \h, 173
 \H, 173
 \j, 173
 kolor, 173, 174, 175, 176
 kolor tła, 176
 \l, 173
 modyfikowanie, 172
 \n, 173
 pusty, 173
 \r, 173
 \s, 173
 \t, 173
 \T, 173
 \u, 173
 \v, 173
 \V, 173
 \w, 173
 \W, 173
 zapisywanie, 177
 zmiana położenia kursora, 176, 177
 zakres, 51, 260, 261, 262
 zestaw, 144
 lokalizacja, 265

PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION



- 1. ZAREJESTRUJ SIĘ**
- 2. PREZENTUJ KSIĄŻKI**
- 3. ZBIERAJ PROWIZJĘ**

Zmień swoją stronę WWW
w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

GRUPA WYDAWNICZA

 **Helion SA**



TWOJA PRZEPUSTKA DO ŚWIATA ZA AWANSOWANYCH UŻYTKOWNIKÓW!

W latach dziewięćdziesiątych ubiegłego stulecia obsługa komputera za pomocą komend wpisywanych w linii poleceń było całkiem naturalne. Ten stan rzeczy uległ zmianie wraz z wprowadzeniem systemu operacyjnego Windows 95, jednak wciąż wiele zadań łatwiej i szybciej można wykonać przy użyciu linii poleceń. Największy potencjał drzemie w linii poleceń systemu operacyjnego Linux. Przekonaj się, jak to wykorzystasz i stać się wirtuozem klawiatury!

Jeżeli jesteś użytkownikiem systemu Linux i pracujesz z wykorzystaniem graficznego interfejsu użytkownika, to najwyższa pora poznać linię poleceń. Sięgnij po tę książkę i daj się ponieść eleganckim komendom, które zebrane w jedną całość potrafią zdziałać cuda. Na samym początku nauczysz się tworzyć pliki, katalogi oraz nawigować po zasobach dysku. W kolejnych rozdziałach przekonasz się, jak pomocne mogą być skrypty powłoki oraz w jaki sposób administrować systemem z poziomu linii poleceń. Ponadto opanujesz obsługę narzędzi do pobierania danych z serwerów FTP, wykorzystasz możliwość przekierowywania strumieni oraz dowiesz się, do czego służy edytor vi.

Ta książka wprowadzi Cię na kolejny poziom wtajemniczenia w obsłudze systemu Linux!

Dzięki tej książce:

- poznasz możliwości linii poleceń systemu Linux
- nauczysz się pisać skrypty automatyzujące pracę
- skorzystasz z zasobów FTP i WWW w trybie tekstowym
- opanujesz obsługę edytora vi

William E. Shotts Jr — znawca systemu Linux. Od 15 lat pracuje z różnymi dystrybucjami oraz programuje w różnych językach. W branży IT działa od ponad 30 lat. Jest twórcą popularnego serwisu LinuxCommand.org.

Helion

32392 numer katalogowy
księgarnia internetowa

<http://helion.pl>

zamówienia telefoniczne

☎ 0 801 339900

☎ 0 601 339900

Informatyka w najlepszym wydaniu

Sprawdź najnowsze promocje:
● <http://helion.pl/promocje>
Książki najchętniej czytane:
● <http://helion.pl/bestsellery>
Zamów informacje o nowościach:
● <http://helion.pl/nowosci>

Helion SA
ul. Kościuszki 1c, 44-100 Gliwice
tel.: 32 230 98 63
e-mail: helion@helion.pl
<http://helion.pl>

sięgnij po WIĘCEJ



KOD KORZYŚCI

ISBN 978-83-283-0174-0



cena: 79,00 zł

