

Słowo wstępne (17)

Przedmowa (19)

Podziękowania (25)

O autorze (27)

Rozdział 1. Pobieranie i instalacja narzędzi oferowanych w trybie open source (29)

- 1.1. Wprowadzenie (29)
- 1.2. Czym jest tryb open source? (30)
- 1.3. Co idea otwartego dostępu do kodu źródłowego oznacza dla nas? (30)
 - o 1.3.1. Odnajdywanie właściwych narzędzi (31)
 - o 1.3.2. Formaty dystrybucji oprogramowania (32)
- 1.4. Wprowadzenie do tematyki plików archiwalnych (33)
 - o 1.4.1. Identyfikacja plików archiwalnych (35)
 - o 1.4.2. Przeglądanie zawartości plików archiwalnych (36)
 - o 1.4.3. Rozpakowywanie plików z pliku archiwalnego (40)
- 1.5. Poznajmy wykorzystywany menedżer pakietów (42)
 - o 1.5.1. Wybór pomiędzy kodem źródłowym a wersją binarną (43)
 - o 1.5.2. Praca z pakietami (46)
- 1.6. Kilka słów o bezpieczeństwie w kontekście pakietów (46)
 - o 1.6.1. Potrzeba uwierzytelniania (48)
 - o 1.6.2. Podstawowe uwierzytelnianie pakietów (49)
 - o 1.6.3. Uwierzytelnianie pakietów z podpisami cyfrowymi (51)
 - o 1.6.4. Podpisy narzędzia GPG, stosowane dla pakietów RPM (52)
 - o 1.6.5. Kiedy uwierzytelnienie pakietu jest niemożliwe (56)
- 1.7. Analiza zawartości pakietu (57)
 - o 1.7.1. Jak analizować pobrane pakiety (59)
 - o 1.7.2. Szczegółowa analiza pakietów RPM (61)
 - o 1.7.3. Szczegółowa analiza pakietów Debiana (62)
- 1.8. Aktualizowanie pakietów (64)
 - o 1.8.1. APT 3 Advanced Package Tool (66)
 - o 1.8.2. YUM 3 Yellowdog Updater Modified (67)
 - o 1.8.3. Synaptic 3 nakładka narzędzia APT z graficznym interfejsem użytkownika (67)
 - o 1.8.4. up2date 3 narzędzie aktualizujące pakiety dystrybucji Red Hat (69)
- 1.9. Podsumowanie (71)
 - o 1.9.1. Narzędzia użyte w tym rozdziale (71)
 - o 1.9.2. Materiały dostępne w internecie (72)

Rozdział 2. Kompilacja kodu źródłowego (73)

- 2.1. Wprowadzenie (73)
- 2.2. Narzędzia kompilujące (74)
 - o 2.2.1. Rys historyczny (74)
 - o 2.2.2. Zrozumieć program make (77)
 - o 2.2.3. Jak przebiega proces łączenia programów (103)
 - o 2.2.4. Zrozumieć biblioteki (104)
- 2.3. Proces kompilacji (109)
 - o 2.3.1. Narzędzia kompilacji GNU (110)
 - o 2.3.2. Etap konfiguracji (skrypt configure) (111)
 - o 2.3.3. Etap kompilacji 3 narzędzie make (113)

- o 2.3.4. Etap instalacji 3 polecenie make install (114)
- 2.4. Zrozumieć błędy i ostrzeżenia (115)
 - o 2.4.1. Typowe błędy w plikach Makefile (115)
 - o 2.4.2. Błędy na etapie konfiguracji (119)
 - o 2.4.3. Błędy na etapie kompilacji (120)
 - o 2.4.4. Zrozumieć błędy kompilatora (124)
 - o 2.4.5. Zrozumieć ostrzeżenia kompilatora (126)
 - o 2.4.6. Zrozumieć błędy programu łączącego (138)
- 2.5. Podsumowanie (140)
 - o 2.5.1. Narzędzia użyte w tym rozdziale (140)
 - o 2.5.2. Materiały dostępne w internecie (141)

Rozdział 3. Szukanie pomocy (143)

- 3.1. Wprowadzenie (143)
- 3.2. Narzędzia pomocy elektronicznej (144)
 - o 3.2.1. Strona man (144)
 - o 3.2.2. Organizacja stron man (145)
 - o 3.2.3. Przeszukiwanie stron man - narzędzie apropos (149)
 - o 3.2.4. Poszukiwanie właściwych stron man - polecenie whatis (151)
 - o 3.2.5. Czego należy szukać na stronach man (152)
 - o 3.2.6. Kilka szczególnie przydatnych stron man (153)
 - o 3.2.7. Narzędzie info projektu GNU (155)
 - o 3.2.8. Przeglądanie stron info (156)
 - o 3.2.9. Przeszukiwanie stron info (159)
 - o 3.2.10. Zalecane strony info (160)
 - o 3.2.11. Narzędzia pomocy uruchamiane na pulpicie (160)
- 3.3. Pozostałe źródła pomocy (162)
 - o 3.3.1. Katalog /usr/share/doc (162)
 - o 3.3.2. Odwołania do innych stron oraz mechanizmy indeksowania (163)
 - o 3.3.3. Zapytania kierowane do pakietów (164)
- 3.4. Formaty dokumentacji (166)
 - o 3.4.1. Formaty TeX, LaTeX i DVI (166)
 - o 3.4.2. Format Texinfo (167)
 - o 3.4.3. Format DocBook (168)
 - o 3.4.4. Język HTML (169)
 - o 3.4.5. Język PostScript (171)
 - o 3.4.6. Format PDF (173)
 - o 3.4.7. Język troff (174)
- 3.5. Źródła informacji w internecie (174)
 - o 3.5.1. Witryna <http://www.gnu.org/> (175)
 - o 3.5.2. Witryna <http://SourceForge.net/> (175)
 - o 3.5.3. Witryna projektu The Linux Documentation Project (176)
 - o 3.5.4. Grupy dyskusyjne Usenet (177)
 - o 3.5.5. Listy dyskusyjne (177)
 - o 3.5.6. Pozostałe fora (178)
- 3.6. Odnajdywanie informacji o jądrze systemu Linux (178)
 - o 3.6.1. Kompilacja jądra (178)
 - o 3.6.2. Moduły jądra (180)
 - o 3.6.3. Pozostałe źródła dokumentacji (182)

- 3.7. Podsumowanie (182)
 - o 3.7.1. Narzędzia użyte w tym rozdziale (182)
 - o 3.7.2. Materiały dostępne w internecie (183)

Rozdział 4. Edycja i konserwacja plików źródłowych (185)

- 4.1. Wprowadzenie (185)
- 4.2. Edytor tekstu (186)
 - o 4.2.1. Edytor domyślny (188)
 - o 4.2.2. Jakich funkcji należy szukać w edytorze tekstu (188)
 - o 4.2.3. Wielka dwójka - vi oraz Emacs (190)
 - o 4.2.4. Vim - udoskonalony edytor vi (191)
 - o 4.2.5. Edytor Emacs (215)
 - o 4.2.6. Atak klonów (227)
 - o 4.2.7. Podstawowe informacje o kilku edytorach tekstu z graficznym interfejsem użytkownika (230)
 - o 4.2.8. Wymagania pamięciowe (235)
 - o 4.2.9. Podsumowanie wiadomości o edytorach (237)
- 4.3. Kontrola wersji (238)
 - o 4.3.1. Podstawy kontroli wersji (238)
 - o 4.3.2. Terminologia obowiązująca w świecie kontroli wersji (240)
 - o 4.3.3. Narzędzia pomocnicze (243)
 - o 4.3.4. Podstawy poleceń diff i patch (243)
 - o 4.3.5. Przeglądanie i scalanie zmian (247)
- 4.4. Upiększacze i przeglądarki kodu źródłowego (254)
 - o 4.4.1. Upiększacze wcięć w kodzie źródłowym (255)
 - o 4.4.2. Artystyczny styl narzędzia astyle (258)
 - o 4.4.3. Analiza kodu za pomocą narzędzia cflow (259)
 - o 4.4.4. Analiza kodu za pomocą narzędzia ctags (262)
 - o 4.4.5. Przeglądanie kodu za pomocą narzędzia cscope (262)
 - o 4.4.6. Przeglądanie i dokumentowanie kodu za pomocą narzędzia Doxygen (264)
 - o 4.4.7. Analiza kodu źródłowego z wykorzystaniem kompilatora (266)
- 4.5. Podsumowanie (268)
 - o 4.5.1. Narzędzia użyte w tym rozdziale (269)
 - o 4.5.2. Bibliografia (270)
 - o 4.5.3. Materiały dostępne w internecie (270)

Rozdział 5. Co każdy programista powinien wiedzieć o jądrze systemu (273)

- 5.1. Wprowadzenie (273)
- 5.2. Tryb użytkownika a tryb jądra (274)
 - o 5.2.1. Wywołania systemowe (276)
 - o 5.2.2. Przenoszenie danych pomiędzy przestrzenią użytkownika a przestrzenią jądra (278)
- 5.3. Mechanizm szeregowania procesów (279)
 - o 5.3.1. Reguły szeregowania procesów (279)
 - o 5.3.2. Blokowanie, wywłaszczanie i rezygnacje (282)
 - o 5.3.3. Priorytety szeregowania i kwestia sprawiedliwości (283)
 - o 5.3.4. Priorytety i wartość nice (287)

- o 5.3.5. Priorytety czasu rzeczywistego (289)
 - o 5.3.6. Tworzenie procesów czasu rzeczywistego (292)
 - o 5.3.7. Stany procesów (294)
 - o 5.3.8. Jak jest mierzony czas pracy procesów (301)
- 5.4. Zrozumieć urządzenia i sterowniki urządzeń (313)
 - o 5.4.1. Rodzaje sterowników urządzeń (314)
 - o 5.4.2. Słowo o modułach jądra (316)
 - o 5.4.3. Węzły urządzeń (317)
 - o 5.4.4. Urządzenia i operacje wejścia-wyjścia (330)
- 5.5. Mechanizm szeregowania operacji wejścia-wyjścia (340)
 - o 5.5.1. Winda Linusa (znana też jako noop) (342)
 - o 5.5.2. Mechanizm szeregowania operacji wejścia-wyjścia z terminem granicznym (343)
 - o 5.5.3. Przewidujący mechanizm szeregowania operacji wejścia-wyjścia (344)
 - o 5.5.4. Mechanizm szeregowania operacji wejścia-wyjścia z pełnym kolejkowaniem (344)
 - o 5.5.5. Wybór mechanizmu szeregowania operacji wejścia-wyjścia (344)
- 5.6. Zarządzanie pamięcią w przestrzeni użytkownika (345)
 - o 5.6.1. Omówienie pamięci wirtualnej (346)
 - o 5.6.2. Wyczerpanie dostępnej pamięci (363)
- 5.7. Podsumowanie (378)
 - o 5.7.1. Narzędzia użyte w tym rozdziale (378)
 - o 5.7.2. Interfejsy API omówione w tym rozdziale (379)
 - o 5.7.3. Materiały dostępne w internecie (379)
 - o 5.7.4. Bibliografia (379)

Rozdział 6. Zrozumieć procesy (381)

- 6.1. Wprowadzenie (381)
- 6.2. Skąd się biorą procesy (381)
 - o 6.2.1. Wywołania systemowe fork i vfork (382)
 - o 6.2.2. Kopiowanie przy zapisie (383)
 - o 6.2.3. Wywołanie systemowe clone (384)
- 6.3. Funkcje z rodziny exec (384)
 - o 6.3.1. Skrypty wykonywalne (385)
 - o 6.3.2. Wykonywalne pliki obiektów (387)
 - o 6.3.3. Różne binaria (389)
- 6.4. Synchronizacja procesów za pomocą funkcji wait (392)
- 6.5. Wymagania pamięciowe procesu (394)
 - o 6.5.1. Deskryptory plików (397)
 - o 6.5.2. Stos (404)
 - o 6.5.3. Pamięć rezydentna i pamięć zablokowana (405)
- 6.6. Ustawianie ograniczeń dla procesów (406)
- 6.7. Procesy i system plików procfs (410)
- 6.8. Narzędzia do zarządzania procesami (413)
 - o 6.8.1. Wyświetlanie informacji o procesach za pomocą polecenia ps (413)
 - o 6.8.2. Zaawansowane informacje o procesach, uzyskiwane z wykorzystaniem formatów (416)
 - o 6.8.3. Odnajdywanie procesów według nazw za pomocą poleceń ps i pgrep (419)

- o 6.8.4. Śledzenie wymagań pamięciowych procesu za pomocą polecenia pmap (420)
- o 6.8.5. Wysyłanie sygnałów do procesów identyfikowanych przez nazwy (422)
- 6.9. Podsumowanie (423)
 - o 6.9.1. Wywołania systemowe i interfejsy API użyte w tym rozdziale (423)
 - o 6.9.2. Narzędzia użyte w tym rozdziale (424)
 - o 6.9.3. Materiały dostępne w internecie (424)

Rozdział 7. Komunikacja pomiędzy procesami (425)

- 7.1. Wprowadzenie (425)
- 7.2. Technika IPC z wykorzystaniem zwykłych plików (426)
 - o 7.2.1. Blokowanie plików (431)
 - o 7.2.2. Wady implementacji techniki IPC z wykorzystaniem plików (432)
- 7.3. Pamięć współdzielona (432)
 - o 7.3.1. Zarządzanie pamięcią współdzieloną za pośrednictwem interfejsu POSIX API (433)
 - o 7.3.2. Zarządzanie pamięcią współdzieloną za pośrednictwem interfejsu System V API (437)
- 7.4. Sygnały (440)
 - o 7.4.1. Wysyłanie sygnałów do procesu (441)
 - o 7.4.2. Obsługa sygnałów (442)
 - o 7.4.3. Maska sygnałów i obsługa sygnałów (444)
 - o 7.4.4. Sygnały czasu rzeczywistego (447)
 - o 7.4.5. Zaawansowane operacje na sygnałach z wykorzystaniem funkcji sigqueue i sigaction (450)
- 7.5. Potoki (453)
- 7.6. Gniazda (454)
 - o 7.6.1. Tworzenie gniazd (455)
 - o 7.6.2. Przykład gniazda lokalnego, utworzonego za pomocą funkcji socketpair (458)
 - o 7.6.3. Przykład aplikacji klient-serwer, zbudowanej z wykorzystaniem gniazd lokalnych (459)
 - o 7.6.4. Przykład aplikacji klient-serwer, zbudowanej z wykorzystaniem gniazd sieciowych (465)
- 7.7. Kolejki komunikatów (466)
 - o 7.7.1. Kolejka komunikatów standardu System V (467)
 - o 7.7.2. Kolejka komunikatów standardu POSIX (471)
 - o 7.7.3. Różnice dzielące kolejki komunikatów standardów POSIX i System V (476)
- 7.8. Semaforey (477)
 - o 7.8.1. Interfejs API semaforów standardu POSIX (483)
 - o 7.8.2. Interfejs API semaforów standardu System V (486)
- 7.9. Podsumowanie (488)
 - o 7.9.1. Wywołania systemowe i interfejsy API użyte w tym rozdziale (489)
 - o 7.9.2. Bibliografia (490)
 - o 7.9.3. Materiały dostępne w internecie (490)

Rozdział 8. Diagnostowanie mechanizmów komunikacji międzyprocesowej za pomocą poleceń powłoki (491)

- 8.1. Wprowadzenie (491)
- 8.2. Narzędzia operujące na otwartych plikach (491)
 - o 8.2.1. Polecenie lsof (492)
 - o 8.2.2. Polecenie fuser (493)
 - o 8.2.3. Polecenie ls (494)
 - o 8.2.4. Polecenie file (495)
 - o 8.2.5. Polecenie stat (495)
- 8.3. Zrzucanie danych z pliku (496)
 - o 8.3.1. Polecenie strings (499)
 - o 8.3.2. Polecenie xxd (500)
 - o 8.3.3. Polecenie hexdump (501)
 - o 8.3.4. Polecenie od (502)
- 8.4. Narzędzia powłoki do obsługi komunikacji międzyprocesowej standardu System V (504)
 - o 8.4.1. Pamięć współdzielona standardu System V (504)
 - o 8.4.2. Kolejki komunikatów standardu System V (507)
 - o 8.4.3. Semaforey standardu System V (509)
- 8.5. Narzędzia powłoki do obsługi komunikacji międzyprocesowej standardu POSIX (510)
 - o 8.5.1. Pamięć współdzielona standardu POSIX (510)
 - o 8.5.2. Kolejki komunikatów standardu POSIX (511)
 - o 8.5.3. Semaforey standardu POSIX (512)
- 8.6. Narzędzia pomocne w pracy z sygnałami (514)
- 8.7. Narzędzia pomocne w pracy z potokami i gniazdami (516)
 - o 8.7.1. Potoki i struktury FIFO (517)
 - o 8.7.2. Gniazda (518)
- 8.8. Identyfikacja plików i obiektów IPC na podstawie i-węzłów (521)
- 8.9. Podsumowanie (523)
 - o 8.9.1. Narzędzia wykorzystane w tym rozdziale (523)
 - o 8.9.2. Materiały dostępne w internecie (523)

Rozdział 9. Doskonalenie wydajności (525)

- 9.1. Wprowadzenie (525)
- 9.2. Wydajność systemu (525)
 - o 9.2.1. Problemy związane z pamięcią (526)
 - o 9.2.2. Wykorzystanie procesora i rywalizacja o dostęp do magistrali (537)
 - o 9.2.3. Urządzenia i przerwania (541)
 - o 9.2.4. Narzędzia umożliwiające identyfikację problemów w zakresie wydajności systemu (550)
- 9.3. Wydajność aplikacji (560)
 - o 9.3.1. Pierwsze kroki - polecenie time (560)
 - o 9.3.2. Zrozumieć architekturę procesora z wykorzystaniem narzędzia x86info (561)
 - o 9.3.3. Stosowanie pakietu Valgrind do analizy efektywności rozkazów (565)
 - o 9.3.4. Wprowadzenie do narzędzia ltrace (570)
 - o 9.3.5. Stosowanie narzędzia strace do monitorowania wydajności programu (572)
 - o 9.3.6. Tradycyjne programy do optymalizacji oprogramowania - gcov oraz gprof (574)

- o 9.3.7. Podstawowe informacje o narzędziu OProfile (583)
- 9.4. Wydajność w środowisku wieloprocessorowym (590)
 - o 9.4.1. Rodzaje systemów SMP (591)
 - o 9.4.2. Programowanie dla komputerów SMP (596)
- 9.5. Podsumowanie (600)
 - o 9.5.1. Omówione w tym rozdziale problemy związane z wydajnością (601)
 - o 9.5.2. Terminy wprowadzone w tym rozdziale (601)
 - o 9.5.3. Narzędzia wykorzystane w tym rozdziale (601)
 - o 9.5.4. Materiały dostępne w internecie (602)
 - o 9.5.5. Bibliografia (602)

Rozdział 10. Diagnozowanie oprogramowania (603)

- 10.1. Wprowadzenie (603)
- 10.2. Najprostsze narzędzie diagnostyczne - funkcja printf (604)
 - o 10.2.1. Problemy związane ze stosowaniem funkcji printf w roli narzędzia diagnostycznego (604)
 - o 10.2.2. Efektywne korzystanie z funkcji printf (610)
 - o 10.2.3. Kilka słów podsumowania metod diagnozowania oprogramowania z wykorzystaniem funkcji printf (620)
- 10.3. Jak opanować podstawy debugera GNU - gdb (622)
 - o 10.3.1. Wykonywanie kodu pod kontrolą debugera gdb (623)
 - o 10.3.2. Zatrzymywanie i wznowianie wykonywania kodu (624)
 - o 10.3.3. Analiza i modyfikowanie danych (636)
 - o 10.3.4. Dołączanie debugera gdb do pracującego procesu (649)
 - o 10.3.5. Diagnozowanie plików rdzenia (649)
 - o 10.3.6. Diagnozowanie programów wielowątkowych za pomocą debugera gdb (653)
 - o 10.3.7. Diagnozowanie zoptymalizowanego kodu (655)
- 10.4. Diagnozowanie obiektów dzielonych (659)
 - o 10.4.1. Kiedy i dlaczego stosujemy obiekty dzielone (659)
 - o 10.4.2. Tworzenie obiektów dzielonych (660)
 - o 10.4.3. Lokalizowanie obiektów dzielonych (661)
 - o 10.4.4. Nadpisywanie domyślnych lokalizacji obiektów dzielonych (662)
 - o 10.4.5. Problemy związane z bezpieczeństwem obiektów dzielonych (663)
 - o 10.4.6. Narzędzia wykorzystywane w pracy z obiektami dzielonymi (663)
- 10.5. Poszukiwanie problemów związanych z pamięcią (667)
 - o 10.5.1. Podwójne zwalnianie pamięci (668)
 - o 10.5.2. Wycieki pamięci (668)
 - o 10.5.3. Przepełnienia buforów (669)
 - o 10.5.4. Narzędzia biblioteki standardowej glibc (671)
 - o 10.5.5. Diagnozowanie problemów związanych z pamięcią za pomocą pakietu Valgrind (675)
 - o 10.5.6. Identyfikacja przepełnień za pomocą debugera Electric Fence (682)
- 10.6. Techniki niekonwencjonalne (685)
 - o 10.6.1. Tworzenie własnych czarnych skrzynek (685)
 - o 10.6.2. Śledzenie wsteczne w czasie wykonywania (688)
 - o 10.6.3. Wymuszanie zrzutów rdzenia (691)
 - o 10.6.4. Stosowanie sygnałów (692)

- o 10.6.5. Diagnozowanie oprogramowania z wykorzystaniem systemu pakietu procfs (693)
- 10.7. Podsumowanie (696)
 - o 10.7.1. Narzędzia wykorzystane w tym rozdziale (697)
 - o 10.7.2. Materiały dostępne w internecie (697)
 - o 10.7.3. Bibliografia (697)

Skorowidz (699)