Systemy Operacyjne 2

Wstęp

Arkadiusz Chrobot

Katedra Systemów Informacyjnych

1 marca 2024

Notatki

Plan Wykładu

Notatki

Literatura

Podstawowa

1. Robert Love, *Jądro Linuksa, przewodnik programisty*, Helion, Gliwice, 2014
2. Jonathan Corbet, Alessandro Rubini, Greg Kroah-Hartman, *Linux Device Drivers*, O'Reilly, 2005
3. Wolfgang Mauerer, *Professional Linux Kernel Architecture*, Wiley Publishing, Inc., Indianapolis, 2008
4. Sreekrishnan Venkateswaran, *Essential Linux Device Drivers*, Prentice Hall, Upper Saddle River, 2008
5. Daniel P. Bovet, Marco Cesati, *Understanding the Linux Kernel, 3rd Edition*, O'Reilly Media, Sebastopol 2005

Notatki

Literatura

Uzupełniająca

1. Mel Gorman, *Understanding the Linux Virtual Memory Manager*, Pearson Education, Inc., Upper Saddle River, 2004
2. Maurice J.Bach, *Budowa systemu operacyjnego UNIX*, WNT, 1995

Notatki

Literatura

Strony www

1. Linux Weekly News
2. Linux Kernel Newbies
3. Linux Cross Reference
4. Kurs tworzenia sterowników dla platformy BeagleBone Black (język angielski)
5. Build own USB device on linux-based board! [en] - Krzysztof Opasiak
6. Linux Kernel Documentation

Notatki

Historia Linuksa

Powstanie systemu operacyjnego Linux, a dokładniej jego jądra, przypada na rok 1991, kiedy prace nad nim zapoczątkował fiński student informatyki, Linus Benedict Torvalds. Jednakże historia związana z tym systemem rozpoczęła się znacznie wcześniej...

Notatki

Historia Linuksa

- W latach 60. ubiegłego wieku MIT, General Electric i AT&T rozpoczęły pracę nad innowacyjnym, wielozadaniowym systemem komputerowym. System operacyjny dla tego komputera, nazwany MULTICS, miały wspólnie przygotować MIT i AT&T.
- Projekt napotkał na szereg problemu, dlatego AT&T się z niego wycofało. Ken Thompson, pracownik Laboratoriów Bella (oddział AT&T) i jeden z programistów MULTICS rozpoczął na komputerze PDP-7 tworzenie gry komputerowej zatytułowanej „Space Travel". W roku 1969 ten projekt przerodził się w prace na systemem operacyjnym Unix (oryginalnie UNICS).
- Stopniowo do tego projektu dołączyli inni programiści zatrudnieni w AT&T, tacy jak Dennis Ritchie, Malcolm Douglas McIlroy, Brian Kernighan, Rob Pike.
- Douglas McIlroy zaproponował koncepcję potoków uniksowych.
- W roku 1973 znaczna część kodu źródłowego Uniksa została przepisana na język C opracowany przez Denisa Ritchie.

Notatki

Historia Linuksa

- Firma AT&T była monopolistą na ryku telekomunikacyjnym w USA. Nie mogła w związku z tym sprzedawać oprogramowania, zatem sprzedawała komputery z zainstalowanym systemem Unix, dostarczając je wraz z jego wersją źródłową.
- Unix stał się bardzo popularny w ośrodkach akademickich i przemyśle. Jego kod źródłowy był wykorzystywany przez uczelnie w ramach zajęć dotyczących systemów operacyjnych. John Lions, australijski informatyk i naukowiec, napisał podręcznik, który zawierał obszerne fragmenty kodu Uniksa.
- Pracownicy Uniwersytetu Kalifornijskiego w Berkeley opracowali i wydali swoją własną wersję Uniksa, która między innymi posiadała implementację gniazd do komunikacji sieciowej. Jednym z tych ludzi był Bill Joy, który dodał obsługę pamięci wirtualnej oraz opracował kilka narzędzi, które stały się bardzo popularne (np. edytor VI, powłoka C).

Notatki

Historia Linuksa

- Wiele innych uczelni i firm zaczęło wydawać swoje własne wersje Uniksa.
- Aby zapanować nad różnorodnością odmian Uniksa, trzy organizacje: IEEE, ISO and The Open Group utworzyły dwa standardy dla tego systemu POSIX i SUS.

Notatki

Historia Linuksa

Mała Dygresja

Przyczyną sukcesu Uniksa był sposób, w jaki został on zaprojektowany. Zbiór zasad projektowych przyjętych przez twórców tego systemu określa się mianem *filozofii Uniksa*. Głównymi jej elementami są:

- dwa najważniejsze pojęcia — proces i plik,
- zasada KISS — „Keep It Simple, Stupid”,
- niewielka liczba wywołań systemowych (około 100),
- proste w użyciu i efektywne wywołania systemowe („wykonuj jedno zadanie, ale wykonuj jej dobrze”),
- wbudowane mechanizmy komunikacji między procesami,
- kod źródłowy zapisany w przenośnym, wysokopoziomowym języku programowania.

Notatki

Historia Linuksa

Kontynuacja

- W latach 80. ubiegłego wieku rząd USA zadecydował o podziale AT&T na wiele mniejszych firm. Jedna z nich przejęła projekty związane z oprogramowaniem i zmieniła licencję Uniksa, również dla wersji, które już zostały sprzedane!
- Nowa licencja zabraniała używania kodu źródłowego Uniksa na zajęciach w uczelniach. Książka Lionsa stała się zakazana!
- Profesor informatyki Andrew S. Tanenbaum z Uniwersytetu Vrije w Amsterdamie, zdenerwowany tą sytuacją opracował system operacyjny MINIX dla studentów, który jest podobny do Uniksa (ang. *Unix-like*), tzn. jest zgodny z POSIX i SUS, ale bazuje na architekturze mikrojądra.
- W 1991 roku, fiński student informatyki, Linus Torvalds, niezadowolony z ograniczeń Miniksa rozpoczął prace nad jądrem swojego własnego systemu operacyjnego i postanowił podzielić się jego kodem z innymi programistami za pośrednictwem serwisu Usenet.

Notatki

Historia Linuksa

Kontynuacja

- Wkrótce dołączyli do niego inni programiści. Narzędzia użytkownika zostały dostarczone przez GNU i inne organizacje.

Notatki

Historia Linuksa

Podsumowanie

Współcześnie Linux jest jednym z najczęściej używanych systemów operacyjnych (być może najczęściej używanym). Kod źródłowy jego jądra liczy 27,8 milionów wierszy (stan na styczeń 2020). Nawet Ken Thompson używa Linuksa.

Notatki

Cechy Linuksa

- Jest darmowym jądrem systemu operacyjnego wydawanym na licencji GNU GPL v. 2.0.
- Jest systemem kompatybilnym z Uniksem (ang. *Unix-like*).
- Większość jądra Linuksa jest napisana w języku C, ale niektóre części są zapisane w asemblerze używającym notacji AT&T.
- Od wersji 6.1 możliwe jest dodawanie do jądra kodu napisanego w języku Rust.
- Jest lub był dostępny dla komputerów bazujących na procesorach ARM, AMD, Intel, POWER, PowerPC, MIPS, Sparc, UltraSparc i wielu innych.
- Wspiera przetwarzanie równoległe bazujące na architekturach SMP i NUMA.
- Jest dostarczane razem z oprogramowaniem użytkowym, takim jak powłoki, X-Window, itd. Tak zbudowany system nazywa się dystrybucją Linuksa.

Notatki

Jądro Linuksa - Przegląd

Jądro jest częścią systemu operacyjnego, która stale rezyduje w RAM działającego komputera. Nadzoruje ono pracę urządzeń i procesów użytkownika. Wszystkie czynności jądra są wykonywane w trybie systemowym procesora. Jądro dysponuje również własnym zbiorem adresów, których używa by uzyskać dostęp się do pamięci operacyjnej. Te adresy, pamięć do której się odwołuje, a także stan jądra i jego zasoby nazywane są *przestrzenią jądra*. Dla odmiany, stan i zasoby procesów użytkownika, razem z zawartością ich pamięci i adresami używanymi od odwoływania do niej są nazywane *przestrzenią użytkownika*. Jądro jest aktywne głównie podczas jego inicjacji, czyli w trakcie uruchamiania systemu komputerowego. Potem staje się oprogramowaniem *reagującym na zdarzenia*. Zdarzenie może być spowodowane przez proces użytkownika poprzez zainicjowanie *wywołania systemowego*. *Wywołanie systemowe* jest specjalną funkcją jądra, która może być wywołana przez proces użytkownika. Zazwyczaj nie odbywa się to bezpośrednio, a za pośrednictwem określonych funkcji należących do standardowej biblioteki języka C.

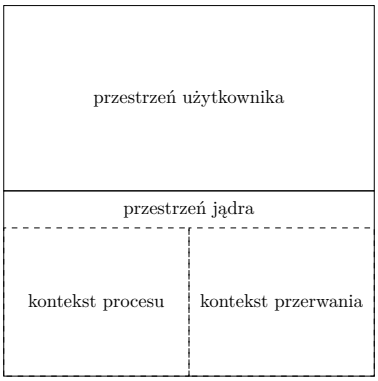
Notatki

Jądro Linuksa - Przegląd

Wywołanie systemowe jest realizowane w przestrzeni jądra i ma dostęp do wszystkich danych o procesie użytkownika, który zapoczątkował jego wykonanie. W związku z tym określa się je jako wykonywane w *kontekście procesu*.
Źródłem zdarzenia obsługiwanego przez jądro może być także urządzenia peryferyjne. Takie zdarzenia nazywane są *przerwaniami* i są asynchroniczne w stosunku do operacji wykonywanych przez jądro. Oznacza to, że mogą one przerwać wykonywanie istotnych czynności, zatem muszą być szybko obsłużone. Z uwagi na to, kod jądra zajmujący się tym działaniem (tak zwana *procedura obsługi przerwania*) jest wykonywany w *kontekście przerwania*, który nie jest związany z żadnym procesem użytkownika. Rysunek 1 ilustruje pojęcia przestrzeni użytkownika, przestrzeni jądra, kontekstu procesu i kontekstu przerwania.

Notatki

Jądro Linuksa - Przegląd



Rysunek 1: Niektóre pojęcia związane z jądrem Linuksa

Notatki

Jądro Linuksa, a jądra innych Uniksów

- Jądro Linuksa, tak jak jądro Uniksa, jest monolityczne.
- Obsługuje dynamicznie ładowane moduły.
- Wspiera architektury równoległego przetwarzania takie jak SMP i NUMA.
- Począwszy od wersji 2.6 obsługuje wywłaszczanie wątków jądra.
- Strony jądra nie podlegają wymianie, w przeciwieństwie do stron procesów użytkownika.
- Nie obsługuje strumieni, które w przypadku Linuksa są całkowicie zrealizowane w przestrzeni użytkownika.

Notatki

Numerowanie wersji jądra Linuksa

Początkowo, kolejne wersje jądra Linuksa były oznaczane przy pomocy trzech liczb rozdzielonych kropką, np: 2.2.1. Pierwsza liczba, to tak zwany *główny numer wersji*. Druga liczba to z kolei *numer poboczny wersji*. W przypadku wersji rozwojowych ten numer był nieparzysty, a dla wersji stabilnych parzysty. W tych pierwszych zachodziły zazwyczaj znaczące zmiany w kodzie i w związku z tym nie były one zalecane do instalacji na środowiskach produkcyjnych. Dla odmiany kod tych drugich nie zmieniał się radykalnie, jedynie wprowadzano do nich poprawki odkrytych defektów. Ostatnia liczba to *numer wydania* (ang. *revision number*), który określał po prostu kolejność wydań.

Notatki

Numerowanie wersji jądra Linuksa

Ten sposób numeracji uległ zmianie wraz z wydaniem wersji 2.6 jądra. Programiści doszli do wniosku, że jest on na tyle dobrze zaprojektowane, że nie będą już potrzebne wersje rozwojowe. Nawet znaczące zmiany mogą być wykonywane na wersji stabilnej. To podejście spowodowało konieczność okresowego stosowania czwartej liczby, która informowała, że wersja zawiera poprawki istotnych defektów.

Notatki

Numerowanie wersji jądra Linuksa

Wraz z wydaniem wersji 3.0 nastąpiła kolejna i jak dotąd ostatnia zmiana w sposobie numeracji. Tym razem numer poboczny wersji oznacza po prostu kolejne wydania. Wersja gotowa do wydania (ang. *release candidate*) jest oznaczana przyrostkiem `-rc`. Pojawiły się również wersje o długim okresie wsparcia (ang. *long term support version*). Na zajęciach laboratoryjnych używana będzie wersja 4.15, utrzymywana przez firmę Canonical. Wykład dotyczy tej wersji, ale także starszych i nowszych. Najnowszą stabilną wersją (stan na luty 2024) jest 6.7.6.

Notatki

Wprowadzenie do programowania w przestrzeni jądra

- Większość kodu jądra jest napisana w język C. Pozostała część jest zaimplementowana w asemblerze i języku Rust.
- Programiści jądra często używają osławionej instrukcji `goto`, zazwyczaj w celach optymalizacji efektywności lub obsługi wyjątków.
- Standardowa biblioteka języka C nie jest dostępna w przestrzeni jądra. Zamiast tego jądro ma zestaw własnych plików nagłówkowych i zamienników większości funkcji dostępnych w C. Na przykład, zamiast funkcji `printf()` używana jest `printk()`. Funkcje przetwarzające łańcuchy znaków także mają swojej odpowiedniki w przestrzeni jądra.
- Kod jądra musi być przenośny, co oznacza że programiści muszą wziąć pod uwagę np. kolejność bitów i bajtów w różnych platformach sprzętowych (ang. *endianness*).
- Problemy związane ze współbieżnością i synchronizacją są zawsze obecne w przestrzeni jądra.

Notatki

Wprowadzenie do programowania w przestrzeni jądra

- Jądro przydziela każdemu procesowi użytkownika stos w swojej przestrzeni adresowej (to jest inny stos, niż ten w przestrzeni użytkownika), który jest używany przez wywołania systemowe i inne funkcje jądra. **Rozmiar tego stosu jest ograniczony do dwóch stron!**
- Działania zmiennoprzecinkowe nie są bezpośrednio obsługiwane w przestrzeni jądra. Na szczęście nie są one zazwyczaj potrzebne.
- Ochrona pamięci w przestrzeni jądra nie jest stosowana.
- Kod jądra korzysta z rozszerzeń języka C opracowanych przez organizację GNU, takich jak funkcje wplatane (ang. *inline*), wstawki asemblerowe, dyrektywy `likely()` i `unlikely()` używane do oznaczania warunków w instrukcjach warunkowych.

Notatki

Pytania

?

Notatki

Dziękuję za uwagę!

Notatki

Notatki

Notatki

Notatki